

Version
1.0

名古屋大学

Autoware ユーザーズマニュアル

Autoware ver.2015.12.12

目次

1. このドキュメントについて	4
2. ROS と Autoware	5
ロボット用のミドルウェア ROS	5
ROS の特徴	6
Autoware	7
3 次元地図の作成と共有	9
自己位置推定 (NDT : Normal Distributions Transform)	9
物体検出	10
経路生成	10
自律走行	10
ユーザインタフェース	10
ROS と Autoware	12
Autoware の構造	13
認知・判断	13
判断・操作・位置推定	14
経路探索	15
3 次元地図などのデータの読み込み(DB、ファイル)	16
ドライバーおよびセンサフュージョン	17
スマートフォン用アプリケーションとのインタフェース	18
その他	18
3. Autoware の主要機能の操作	19
準備	19
デモデータの準備	19
Runtime Manager の起動	20
rviz の設定	21
Quick Start での実行	21
PointCloud のロード (Quick Start)	21
ドライバのロード (Quick Start)	21
Autoware 主要機能	22
自己位置推定 (NDT : Normal Distributions Transform)	22
物体検出 (Detection)	23
経路計画	24
軌跡計画	25
ダイナミックマップ	26
AutowareRider	29
AutowareRoute	29
主要機能	29
Autoware Rider の起動	30
経路データ生成アプリケーションの使用方法	31
ROS PC への経路データ転送手順	32
CAN データ収集アプリケーションの使用方法	32

ROS PC への CAN データ転送手順.....	33
Launch ファイルの起動方法	33
4. Autoware の画面解説	34
Runtime Manager	34
Runtime Manager – QuikStart タブ	35
Runtime Manager – Setup タブ	37
Runtime Manager – Map タブ	38
Runtime Manager – Sensing タブ	40
Runtime Manager – Computing タブ	43
Runtime Manager – Interface タブ	47
Runtime Manager – Database タブ	49
Runtime Manager – Simulation タブ	51
Runtime Manager – Status タブ	53
runtime manager - Topics タブ	55
ROSBAG Record ダイアログ	56
rviz	58
AutowareRider	59
AutowareRoute	64
5. 環境構築.....	66
インストール	66
OS	66
ROS.....	67
Velodyne ドライバ.....	68
OpenCV.....	68
Qt (必要な場合)	69
CUDA (必要な場合)	70
FlyCapture (必要な場合)	71
Autoware	72
AutowareRider.....	73
canlib	74
SSH の公開鍵の作成.....	74
6. 用語.....	75
7. 関連文書	78

チャプター
1

1. このドキュメントについて

ドキュメントの位置付けと、ドキュメント内で使用されるアイコンについて説明します。

名古屋大学が無償で提供する Autoware に関するドキュメントは、以下の2つです。

「Autoware ユーザーズ マニュアル」

「Autoware デベロッパーズ マニュアル」

上記ドキュメントには日本語版です。今後、それぞれの英語版が作成される予定です。

「Autoware User's Manual」

「Autoware Developer's Manual」

2. ROS と Autoware

Autoware の操作をする前に、理解を深めるために *ROS* と *Autoware* について解説します。

ロボット用のミドルウェア ROS

近 年、ロボットの多様な可能性が着目されており、専門家以外からのロボット開発参入によって、技術力だけでなく、様々な分野への発展へつなげることができます。しかし、ロボット開発はその機能の高度化と複雑化により、より難易度が増しています。パソコンやスマートフォンと異なり、それぞれのハードやOS、プログラム言語が異なることなども、ロボットの専門家はもとより、ロボット開発者の新規参入を妨げる大きな理由となっていました。

のことから、共通プラットフォーム化への期待が高まり、様々なプラットフォームが作成されています。プラットフォームが共通となることで、他者が作成したソフトウェアを部品のように組み合わせ、再利用することで開発期間を短縮し、開発者が各自の得意とする分野研究をより探究できるようになることが期待できます。

ROS (Robot Operating System) は米 Willow Garage が開発し、OSRF (Open Source Robotics Foundation) が維持・管理している、ロボットソフトウェア開発のためのソフトウェアフレームワークです¹。オープンソース化された ROS は欧米を中心に、日本国内でも普及はじめています。

ROS は名前に「OS」と付いてはいますが、Windows や Linux のような OS とは異なり、Unix ベースの OS 上で動作するミドルウェアです。

¹ https://en.wikipedia.org/wiki/Willow_Garage

ROS の特徴

① ライブラリとツールの提供

ROS ではロボット用のソフトウェアを開発する為のライブラリとツールが提供されます。

以下は主なライブラリとツールです。

- ✓ 独自のビルドシステム (Catkin)
- ✓ 画像処理ライブラリ (OpenCV)
- ✓ データロギングツール (rosbag)
- ✓ データとソフトウェア状態の視覚化ツール (rviz)
- ✓ 座標変換ライブラリ (tf)
- ✓ Qtベースの GUI ソフト開発ツール (tqt)

② プロセス間通信

ROSはモジュール間接続および連携のフレームワークに「Topic」を用いた Publish /Subscribe 型のメッセージパッシングを用いています。メッセージパッシングとは 1つ以上の受信者に対して送信者がデータを配信するプロセス間通信の方式です。この方式により分散型システムへの拡張ができます。

ROS では「Node」と呼ばれるプロセスを複数立ち上げ、各 Node が独立して実行されます。

Node 間では Topic に「Message」と呼ぶデータを書き込み (Publish) し、その Topic を購読している Node が書き込まれた Message を読み込み (Subscribe) ます。

③ 構成要素

- ・バッグファイル (rosbag)

ROS では全ての Topic 上のメッセージがタイムスタンプと共に「rosbag」と呼ぶ.bag ファイルに記録でき、rviz で記録時と同じタイミングで再生することができます。多様なセンサを用いるロボット開発では、各種センサの動作情報の時間を同期させて総合的に解析するのは難しいため、ROS ではこれによって効率的に不具合などの解析ができます。また、記録した処理を繰り返し入力することもできるので、カメラやセンサが無くてもデバッグすることができます。

- ・Launch ファイル

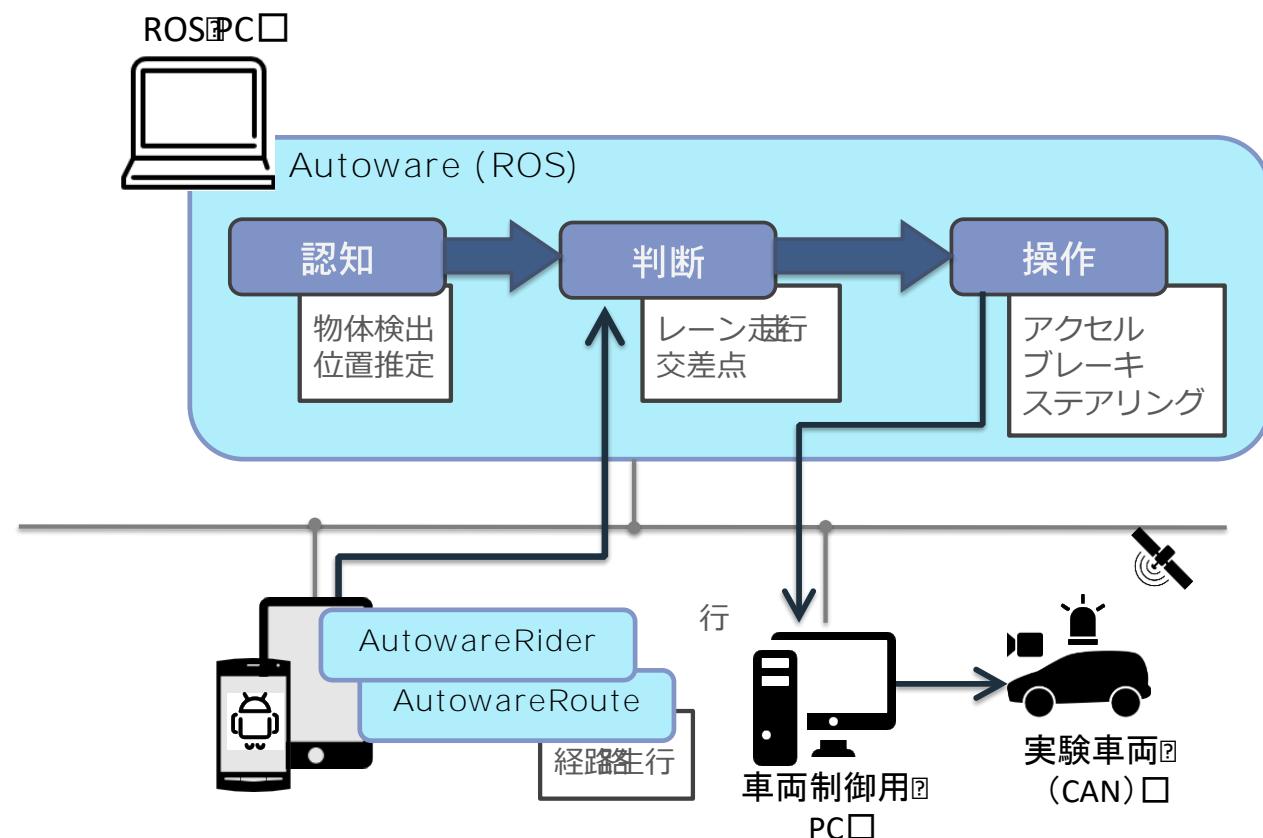
複数の Node を一度に起動するには、Launch ファイルを用意してそれを実行することができます。Launch ファイルには起動したい Node の情報など起動に必要な情報を XML 形式で記述します。

Autoware

A utoware は、ROS 上で動くオープンソースソフトウェアです。自動運転技術の研究開発用に GitHub 上で公開しています。

自動運転システムの多くは「認知」、「判断」、「操作」で構成されています。

Autoware では 3 次元地図生成、自己位置推定、物体検出、走行制御といった、自動運転に必要な機能を提供しています。



Autoware では、LIDAR や車載カメラを組み合わせて自車位置の推定をし、LIDAR と GNSS を用いて歩行者、周囲の車両、信号機などの周囲の物体検出を行ないます。

レーンや交差点での走行・停止といった「判断」には組込みメニーコア CPU を用いて実現しています。

実際の車両の「操作」にあたる車両制御については従来の自動車に搭載されている機能を用い、運転支援や安全診断などの支援系は組込みメニーコア CPU を用いて実現しています。

3次元地図の作成と共有

Autoware の最大の特徴は 3 次元地図を基に自動運転を行う点にあります。3 次元地図とは、一般的なカーナビゲーションなどで使われている 2 次元の地図とは違い、道路周辺に設置されている立体物などを含めてさまざまな情報が取り込まれた地図です。今後、自動運転技術を実用化するには、3 次元地図の普及が重要な役割を果たします。Autoware では、自車位置の推定は Autoware を動作させる PC にあらかじめ取り込んでおいた 3 次元地図と、車両上部に設置した LIDAR が取得する車両周辺の情報を照らし合わせることで、自車の位置を把握する仕組みになっています。その精度は約 10cm と、GPS よりもはるかに高精度です。

2015 年 9 月時点では、3 次元地図があるのはごく一部の地域だけです。もし Autoware を搭載した自動運転車が 3 次元地図が用意されていない区域に入った場合には、LIDAR が取得する情報から、リアルタイムで新たに 3 次元地図を生成することができます。

このようにして生成された 3 次元地図は、Autoware を使用して開発する開発者自身の自動運転技術開発のために活用するだけでなく、Autoware の機能を用いて名古屋大学のサーバにアップロードして共有することができます。この機能により、3 次元地図のオンライン更新も実現できます。この共有の仕組みによって 3 次元地図の作製専用車では入れないような細かな路地など、市街地のすみずみまで 3 次元地図を用意できるようになります。

3 次元地図から地物データを抽出することで、ベクタ形式の 3 次元地図を生成することもできます。

自己位置推定 (NDT : Normal Distributions Transform)

自車位置は、Point Cloud と LIDAR データを入力として、NDT アルゴリズムをベースとしたスキャンマッチングを行うことで、自車位置を 10cm 程度の誤差で推定することができます。

物体検出

カメラ画像を入力として、DPM（Deformable Part Models）アルゴリズムによる画像認識を行うことで、車や歩行者、信号機を検出できます。カルマンフィルタを利用したトラッキングも可能で、検出精度を高めています。また、3次元 LIDAR データをfusion（融合）することで検出した物体の距離も取得できます。

信号機も検出できます。自己位置推定の結果と高精度 3次元地図から信号機の位置を正確に算出し、その3次元位置をセンサフュージョンによってカメラ画像上に射影し、そこから画像処理によって色判別することで信号機検出を実現しています。

経路生成

目的地までのルートは AutowareRoute (MapFan を使用した経路データ生成アプリケーション) で生成し、自動運転システムが、そのルート上で使用する車線を決定します。経路データ生成アプリケーションは、スマートフォンのカーナビアプリケーションとして提供されています。車線内では運動学的に導かれた軌跡を生成します。

自律走行

生成した経路には適切な速度情報も含まれており、その速度を目安に自律走行します。また、経路には 1m 間隔で設定された目印の「way point」が設定しており、それを追っていくことで経路追従を行います。カーブでは近くの way point、直線では遠くの way point を参照することで、自律走行を安定化しています。経路から逸脱した場合は、近傍の way point を目指して経路に戻ります。もっとも安全な経路を選択して走行します。

ユーザインターフェース

Autoware のデベロッパ用ユーザインターフェースである「Runtime Manager」を利用して、位置推定や物体検出、経路追従などの処理を簡単に操作できます。

`rviz` を用いて 3 次元地図上での自己位置推定、物体検出、経路生成、経路追従を統合し、可視化できます。

Autoware のユーザ用タブレットユーザインタフェースである「AutowareRider」を利用すると、タブレットでナビや経路生成、自動運転モードへの移行などの処理を簡単に操作できます。

自動運転システムが使っている3次元地図情報を可視化し、車載ディスプレイやOculus デバイスに投影することもできます。

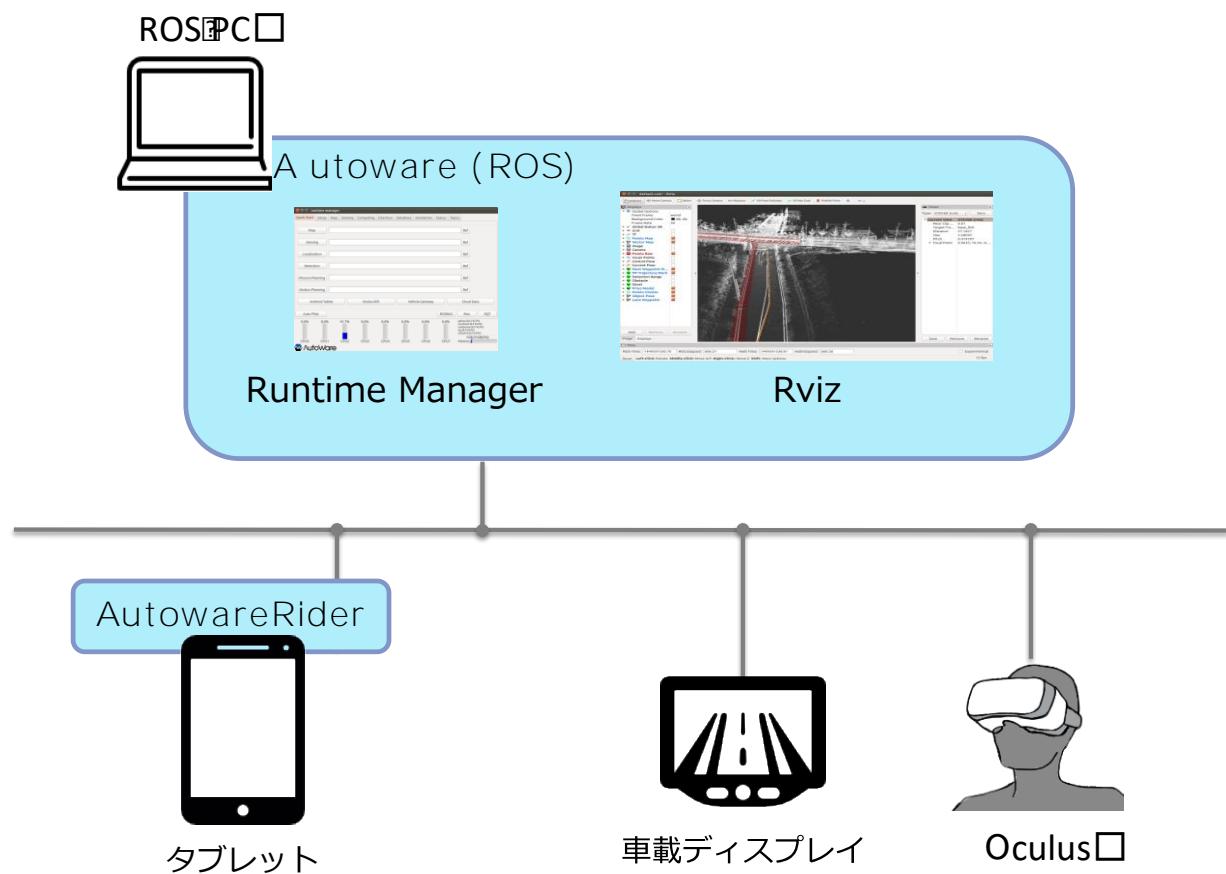


図 2 Autoware のユーザインタフェース

ROS と Autoware

R OS と Autoware の構成は以下のようになっています。

ROS は Unix ベースのプラットフォームでのみ、動作します。

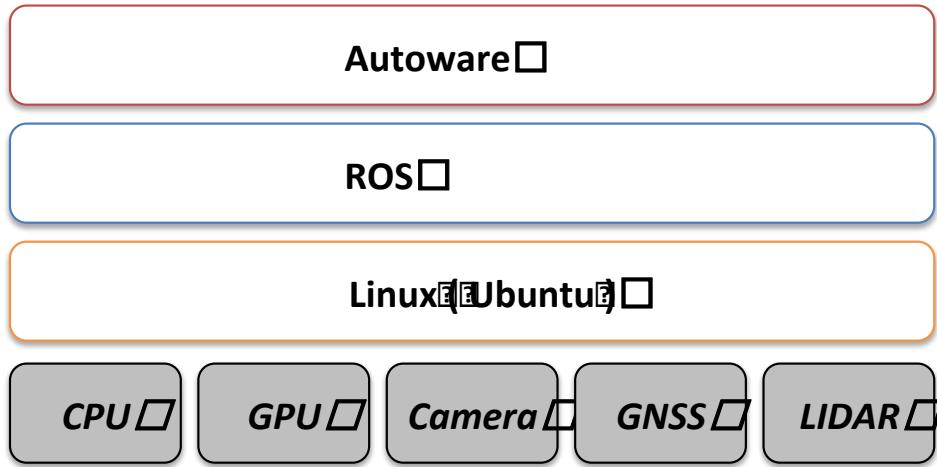


図 3 ROS と Autoware のモデル

Autoware の構造

認知・判断

ros/src/computing/perception/detection
[road_wizard](#), [cv_tracker](#), [lidar_tracker](#)

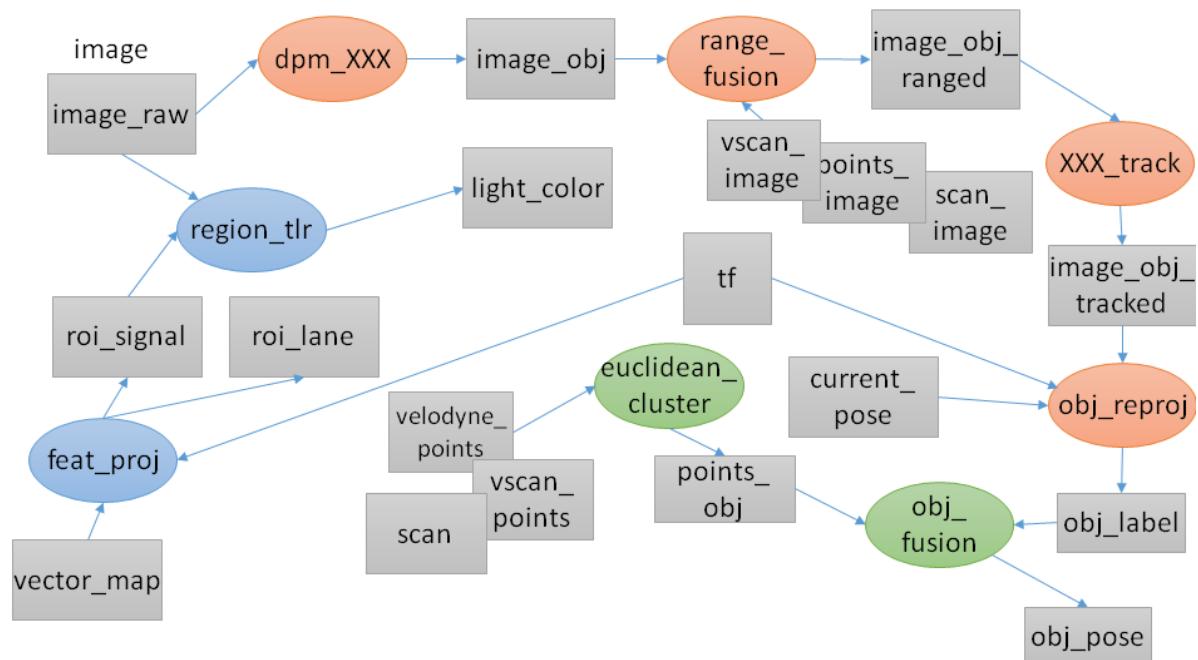


図 4 認知・判断

判断・操作・位置推定

ros/src/computing/perception/localization

`ndt_localizer, orb_localizer, gnss_localizer`

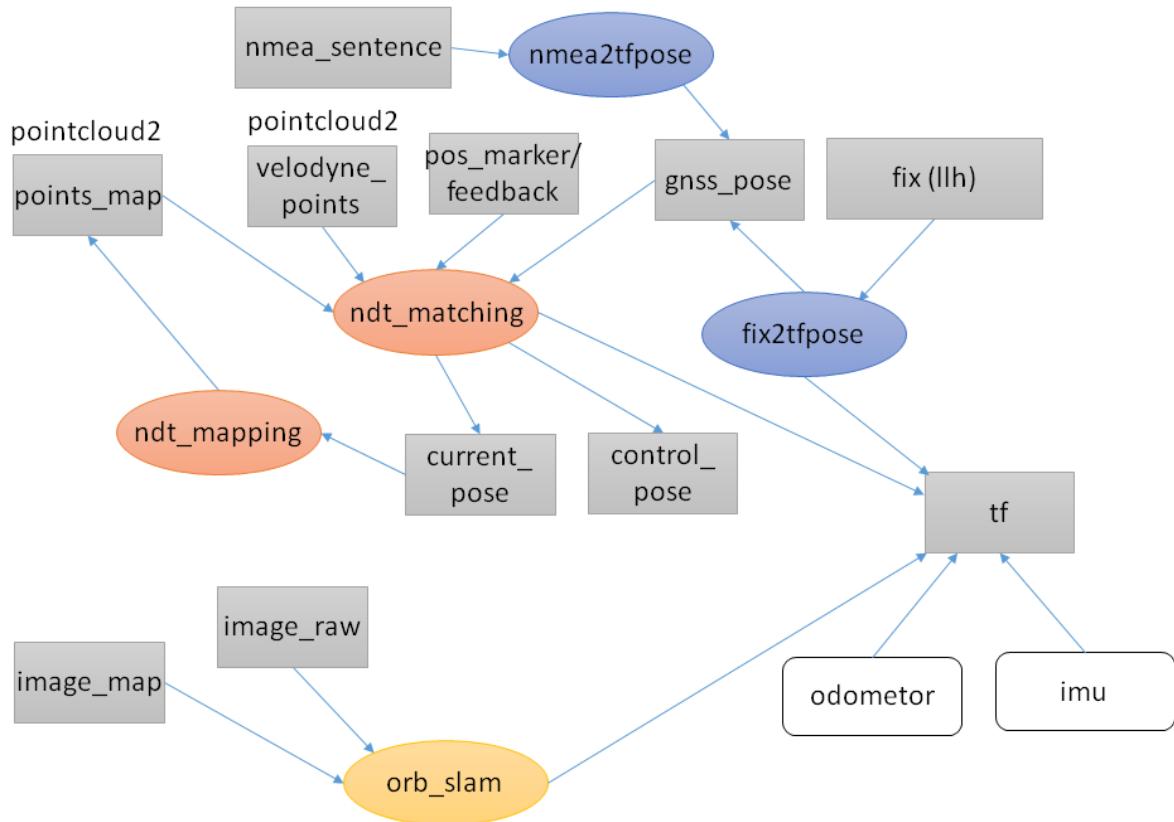


図 5 判断・操舵・位置推定

経路探索

ros/src/computing/planning

lane_planner, driving_planner, waypoint_maker, waypoint_follower

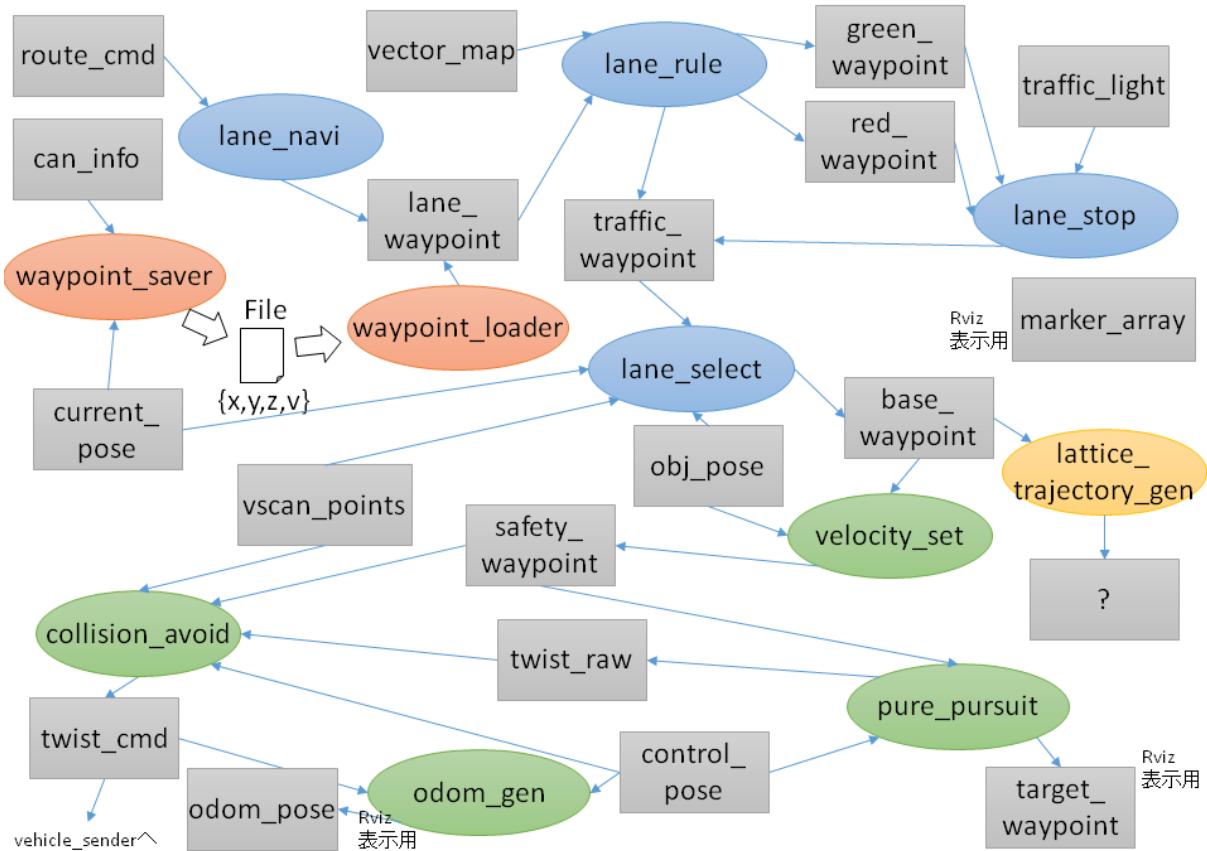


図 6 経路探索

3次元地図などのデータの読み込み(DB、ファイル)

ros/src/data

map_db, map_file, pos_db, dynamic_map

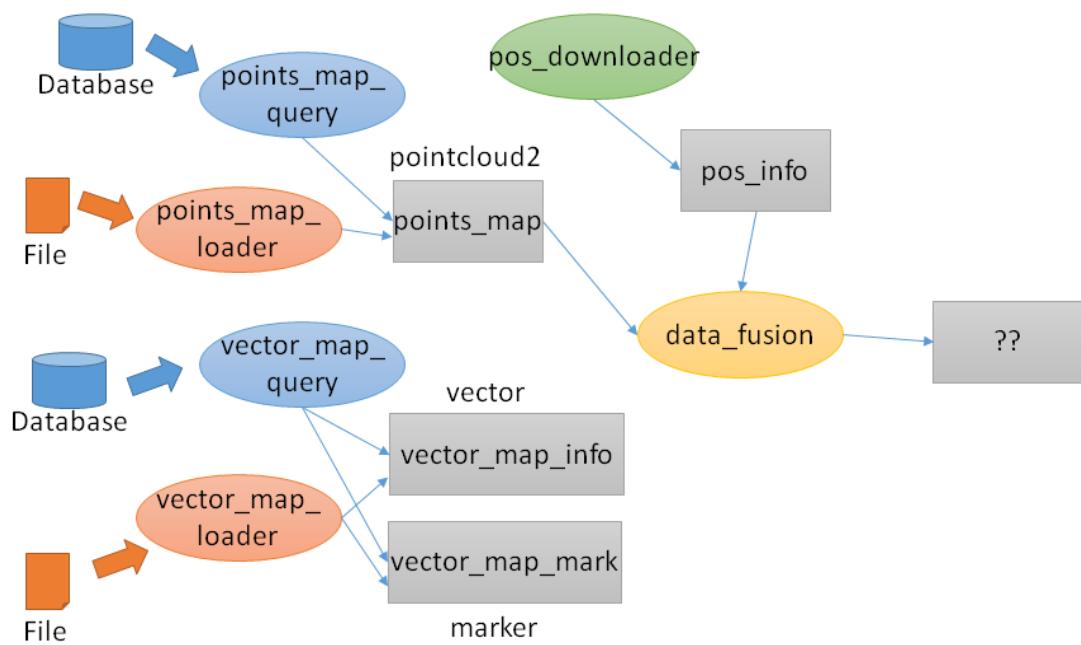


図 7 3次元地図などのデータの読み込み

ドライバおよびセンサフュージョン

ros/src/sensing/drivers および ros/src/sensing/fusion

Drivers and Fusion

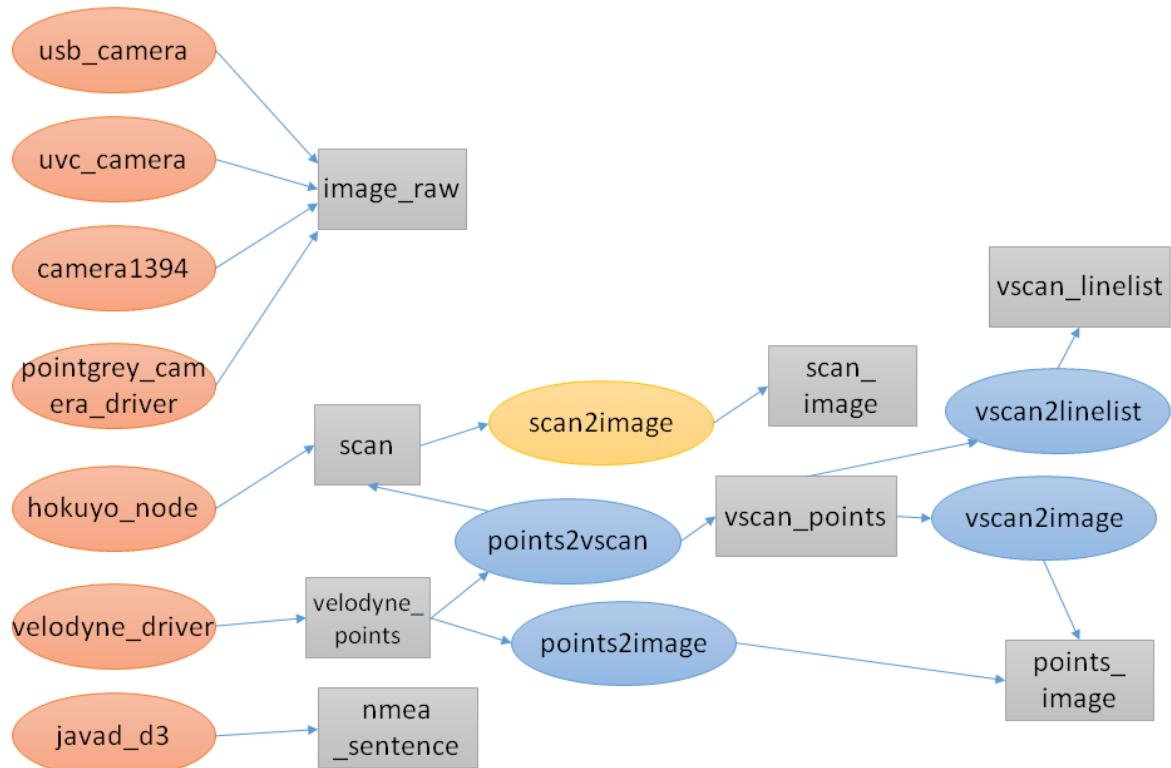


図 8 ドライバおよびセンサフュージョン

スマートフォン用アプリケーションとのインターフェース

ros/src/socket

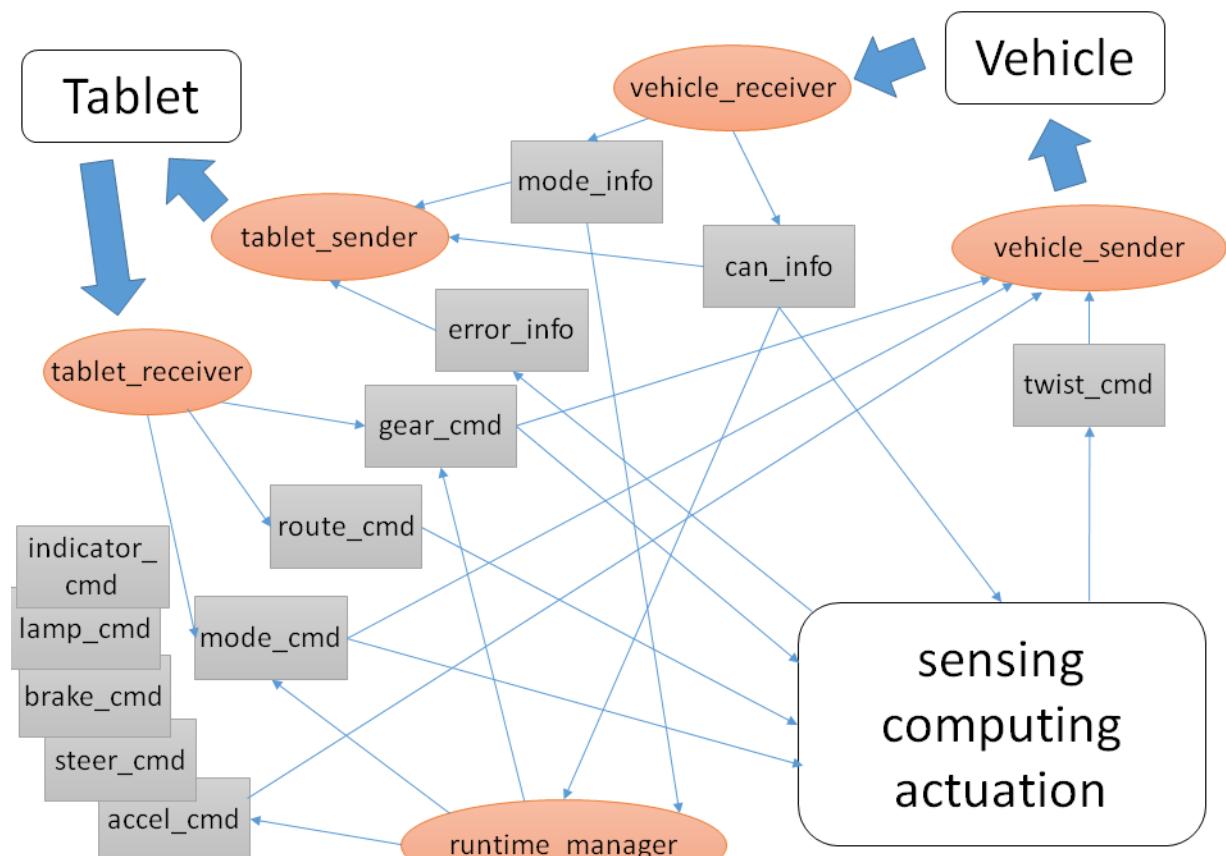


図 9 スマートフォン用アプリケーションとのインターフェース

その他

ros/src/util/

Runtime Manager、サンプルデータ、擬似ドライバなど

ui/tablet/

スマートフォン用アプリケーション

vehicle/

車両の制御、情報取得など

チャプター 3

3. Autoware の主要機能の操作

いよいよ *Autoware* の主要な機能を操作します。Runtime Manager から操作をします。ユーザインターフェース画面の詳細については、チャプター4 を参照してください。

準備

Runtime Manager の「Quick Start」タブからは簡単な操作で Autoware の主要な機能を実行することができます。名古屋大学では Quick Start で使えるデモデータを用意しています。ここではそのデモデータをダウンロードして準備する方法について説明します。

デモデータの準備

Runtime Manager の「Quick Start」タブからの操作を説明します。

ここでは、必要なデータをディレクトリ「`~/.autoware/data`」に配置されているとして説明します。

① デモデータのダウンロード

デモデータを以下からダウンロードし、「`~/.autoware/data`」に配置します。

- ✧ デモ用の launch ファイルを生成するスクリプト
http://db3.ertl.jp/autoware/sample_data/my_launch.sh
- ✧ デモで使うデータ(守山地区の地図・キャリブレーション・経路)
http://db3.ertl.jp/autoware/sample_data/sample_moriyama_data.tar.gz
- ✧ ROSBAG データ
http://db3.ertl.jp/autoware/sample_data/sample_moriyama_150324.tar.gz

注) この ROSBAG データには画像情報が含まれていないため、物体検出(Detection)はできません

② デモデータの展開

ダウンロードしたデモデータを 「~/.autoware/」 以下に展開します。

```
$ tar xfz sample_moriyama_data.tar.gz -C ~/.autoware/
```

③ スクリプトの実行

以下のスクリプトを実行して、「Quick Start」 タブからデモを実行するための launch ファイルを生成します。

```
$ sh my_launch.sh
```

④ 実行すると、以下の launch ファイルが生成されます。

my_launch/	
my_map.launch	# 地図のロード
my_sensing.launch	# ドライバのロード
my_localization.launch	# 位置認識
my_detection.launch	# 物体検出
my_mission_planning.launch	# 経路計画
my_motion_planning.launch	# 軌跡計画

⑤ 別ディレクトリに生成したい場合

データを 「~/.autoware/data」 以外の場所に配置する場合は、シェルスクリプトを実行する際に、引数にデータを配置したディレクトリの指定を行って下さい。

例) ~/.autoware/data/quick_start/rosbag_sample/ にデータを配置した場合

```
$ sh my_launch.sh ~/.autoware/data/quick_start/rosbag_sample/
```

Runtime Manager の起動

ROS PC で Autoware/ros/run をダブルクリック、もしくは ./run で起動できます。

run ファイルにはシェルスクリプトが記載されています。

run を実行すると、端末（ターミナル）が 2 つ立ち上がります。

1 つは「rscore」、もう 1 つは runtime manager の出力結果が表示される端末です。

rviz の設定

- ① 「Runtime Manager」の右下の [Rviz] ボタンを押下して rviz を起動します。
- ② 「Rviz」のメニュー [File] - [Open Config] を選択します。
- ③ [Choose a file to open] ダイアログが表示されます。以下の Config ファイルを選択して [Open] ボタンを押下します。

Autoware/ros/src/.config/rviz/default.rviz

Quick Start での実行

以降は Runtime Manager の「Quick Start」タブから Autoware の主要機能を簡単に実施する手順です。

PointCloud のロード（Quick Start）

- ① PointCloud のロードをします。[Quick Start] タブの [Map] のファイル選択ダイアログに デモデータの準備 で生成した “my_map.launch” を指定して [Map] ボタンを押下します。

ドライバのロード（Quick Start）

- ① [Quick Start] タブの [Sensing] のファイル選択ダイアログに デモデータの準備 で生成した “my_sensing.launch” を指定して [Sensing] ボタンを押下します。

Autoware 主要機能

自己位置推定 (NDT : Normal Distributions Transform)

rosbag を使用する場合の手順を説明します。

- ① [Simulation] タブの[Clock] と [Sim Time] をチェックして On にします。 [Sim Time] を有効にしていると、この時点では地図は表示されません。既に[Map]ボタンを押して起動状態になっている場合は一旦 [Map] ボタンを押して終了し、再度 [Map]ボタンを押下して起動し直してください。
- ② [Map] ボタンを押下して PointCloud およびベクタ地図をロードします。
- ③ [Computing] タブの [mdt_matching] - [app]をクリックしてダイアログを開き、[GNSS] にチェックが入っている事を確認した上で[OK] ボタンを押下します。
- ④ 地図のロードをします。[Quick Start] タブの [Localization] のファイル選択ダイアログにデモデータの準備で生成した my_localization.launch を指定して [Localization]ボタンを押下します。
- ⑤ [Simulation] タブのファイル選択ダイアログで、実行する rosbag ファイルを指定し、[Play] ボタンを押下します。rosbag のプレイが開始すると地図が表示されます。NDT が実行されるとそれらも表示されます。表示されない場合は rviz の [Reset] ボタンを押下、または、[Displays] の一覧にある「Points Map」と「Vector Map」のチェックを外し、再びチェックを入れるなどの操作をしてください。
- ⑥ 自己位置推定の結果がGPSの矢印に追従しない場合は、rvizの上部にある [2D Pose Estimate] をクリックして GPSの矢印の付近にカーソルを合わせてクリックして下さい。

物体検出 (Detection)

- ① [Qutick Start] タブの [Detection] のファイル選択ダイアログにデモデータの準備で生成した「my_detection.launch」を指定して [Detection] ボタンを押下します。
- ② NDT 実行中に物体検出を行なって成功すると、車両は青い球、歩行者は緑の球で表示されます。
注) デモデータの ROSBAG データには画像情報が含まれていないため、物体検出はできません

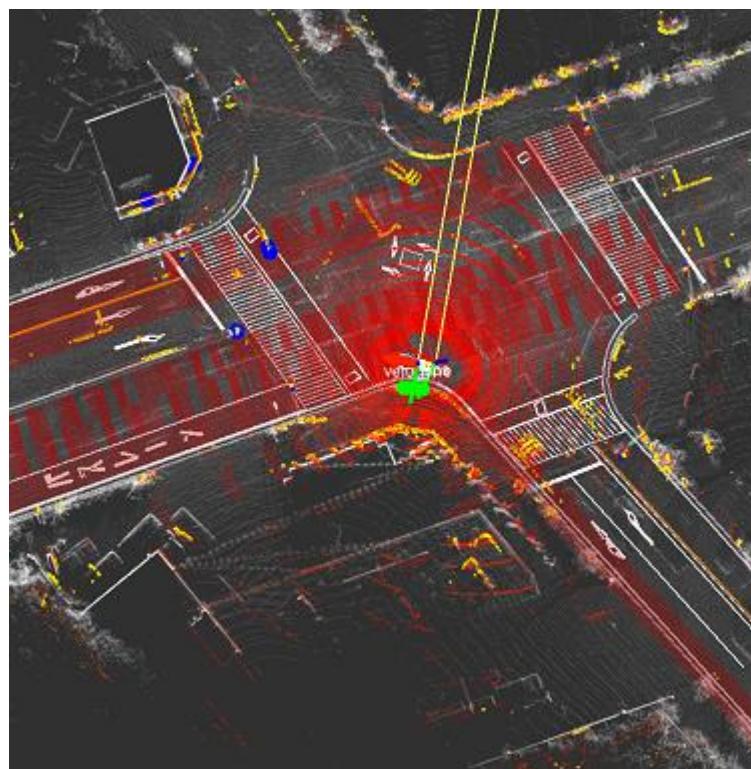


図 10 物体検出

経路計画

- ① [Qutick Start] タブの [Mission Planning] のファイル選択ダイアログに デモデータの準備 で生成した 「my_mission_planning.launch」 を指定して [Mission Planning] ボタンを押下します。
 - ② 実行すると、青い線でパスと速度が表示されます。

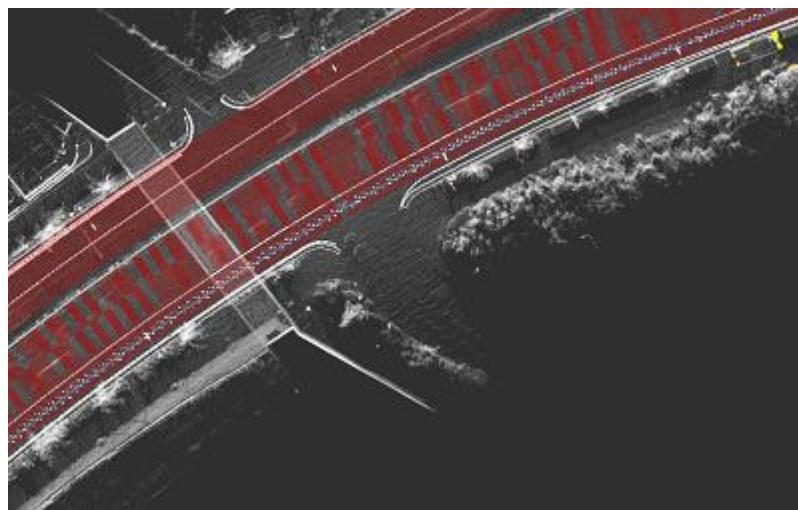


図 11 経路探索

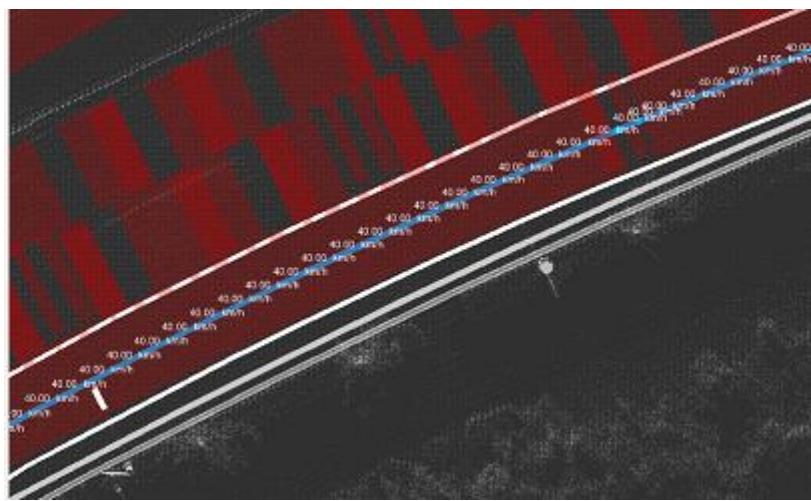


図 12 経路探索(拡大)

軌跡計画

- ① [Computing] タブの [pure pursuit] - [app] をクリックしてダイアログを開き、[Waypoint] にチェックが入っている事を確認した上で OK ボタンを押します。
- ② [Quick Start] タブの [Motion Planning] のファイル選択ダイアログに デモデータの準備 で生成した 「my_motion_planning.launch」 を指定して [Motion Planning] ボタンを押下します。
- ③ 経路計画で設定したパスが表示されているところまで来ると、パス上に青い球、Pure Pursuit により計算される赤い円が表示されます。

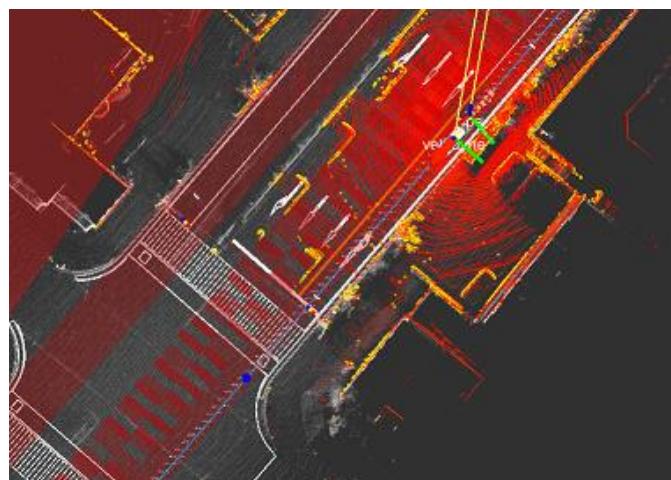


図 13 軌跡計画

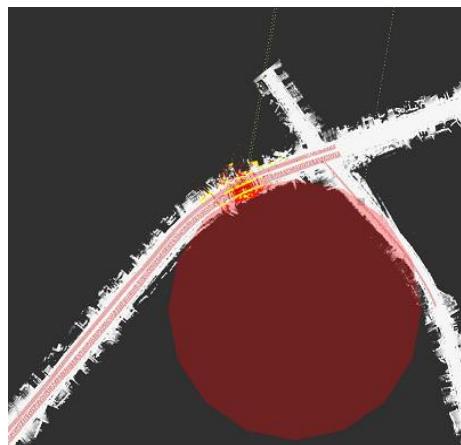


図 14 軌跡計画(拡大)

ダイナミックマップ

自車や他車が認識した車および人の情報を、名古屋大学のデータベースを使って共有する方法です。

- ① 以下のトピックが配信されている状態にします(tmp)。

すべて必須というわけではなく、以下が配信されていれば、その情報をデータベースに登録します。

current_pose (自車, ndt_matching が配信)

obj_car_pose (他車, obj_fusion が配信)

obj_person_pose (人, obj_fusion が配信)

- ② 情報を提供する側で [Database] タブの [Position] – [pos_uploader] - [app] をクリックし、データベースサーバに SSH でアクセスするための情報を入力し、[OK] ボタンを押します。

(SSH 鍵の生成手順は、環境構築の手順にあります。)

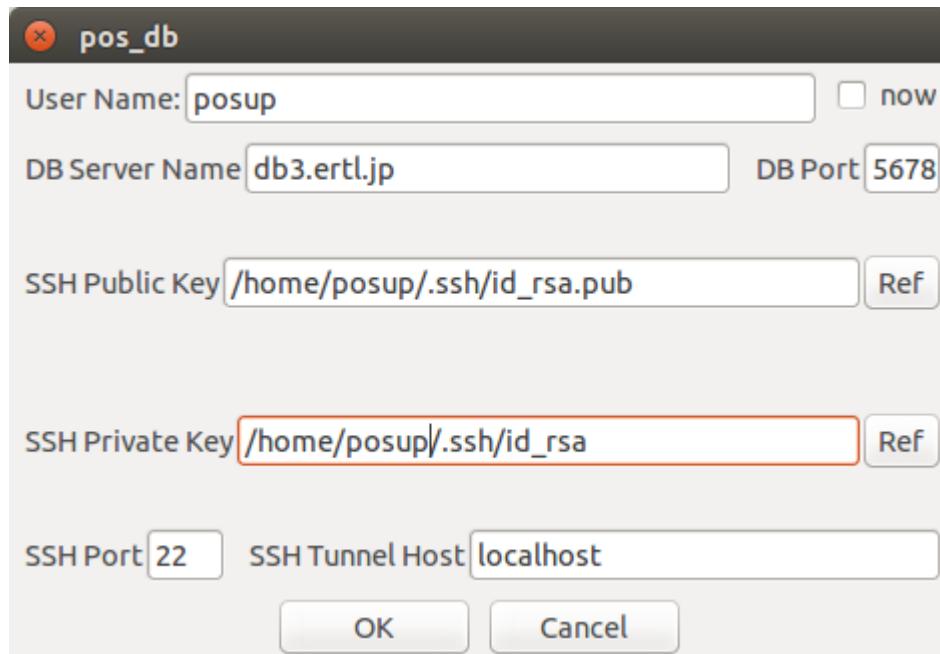


図 15 pos_db

③ 情報を提供する側で [Database] タブの [Position] – [pos_uploader] にチェックします（ノードを起動します）。

④ 情報を閲覧する側で [Database] タブの [Position] – [pos_uploader] - [app] をクリックし、データベースサーバに SSH でアクセスするための情報を入力し、OK ボタンを押します。show my pose にチェックすると、自車の位置を配信します。

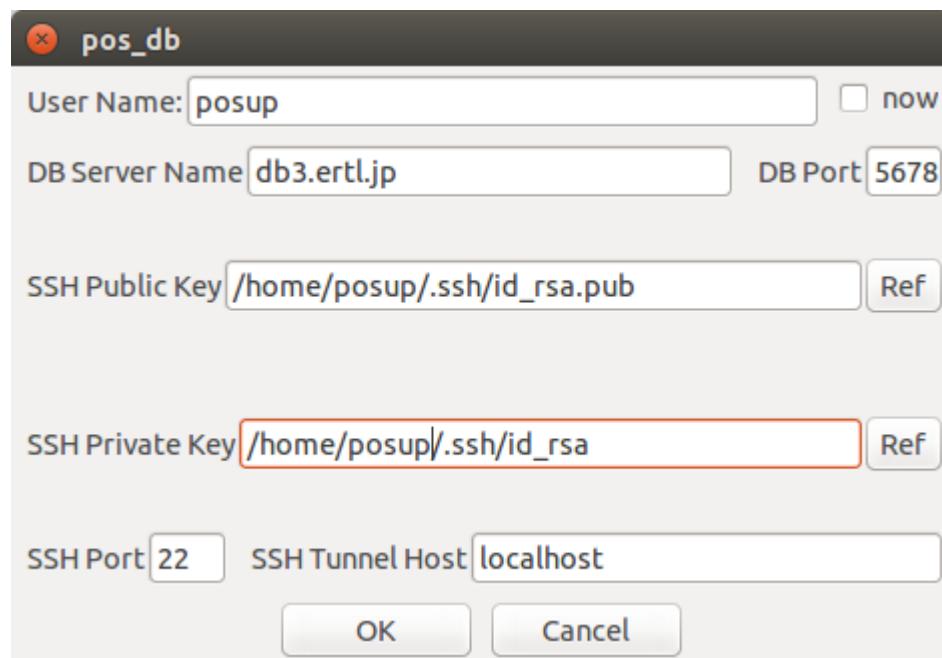


図 16 pos_db

⑤ 情報を閲覧する側で [Database] タブの [Position] – [pos_downloader] にチェックします。

⑥ 情報を閲覧する側で rviz で、トピック 「/mo_marker」 の 「Marker」 を追加すると、認識した自車、他社および人が表示されます。

⑦ データベース(VoltDB)に作成済のテーブル「can」は、以下の通りです。

表 1 can テーブル

列名	型	概要
id	varchar(32)	MAC アドレスや端末固有情報など
lon	float	緯度 (Android 端末)
lat	float	経度 (Android 端末)
h	float	未使用
x	float	平面直角座標 x (pos_uploader)
y	float	平面直角座標 y (pos_uploader)
z	float	平面直角座標 z (pos_uploader)
area	float	平面直角座標の系番号 (pos_uploader, 7 固定)
dir	float	方向 (Android 端末)
acct_x	float	加速度 x (pos_uploader)
acct_y	float	加速度 y (pos_uploader)
acct_z	float	加速度 z (pos_uploader)
vec	float	未使用
type	smallint	1=自車, 2=認識した車, 3=認識した人, 0=Android 端末の位置
self	smallint	未使用
tm	timestamp	タイムスタンプ(GMT)

AutowareRider

AutowareRider は、ROS PC で動作する Autoware を、タブレット端末から操作するための Android アプリケーションです。TV ドラマ「ナイトライダー」に似た UI を持ちます。

AutowareRoute

AutowareRoute は、MapFan SDK で実装された、経路データ生成のための Android アプリケーションです。

主要機能

AutowareRider は、以下の機能を提供します。

- ✓ AutowareRoute で生成した経路データを ROS PC へ送信
- ✓ CAN データ収集アプリケーションを起動
- ✓ ボタン操作で ROS PC の Launch ファイルを起動
- ✓ ROS PC から受信した CAN データを UI へ反映

Autoware Rider の起動

- ① ROS PC で Runtime Manager を起動します。
- ② Main タブ[Network Connection] - [Tablet UI]の Active ボタンを押し、以下を起動します(tmp)。

tablet_receiver
tablet_sender

- ③ [Computing]タブの[Planning] - [Path]の各[app]から、以下を設定します。

[lane_navi]
vector_map_directory
高精度地図が格納されたディレクトリ

[lane_rule]
vector_map_directory
高精度地図が格納されたディレクトリ
ruled_waypoint_csv
waypoint が保存されるファイル

Velocity
速度 (単位: km/h、初期値: 40、範囲: 0~200)

Difference around Signal
信号の前後で加減速する速度 (単位: km/h、初期値: 2、範囲: 0~20)

[lane_stop]
Red Light
赤信号時の速度へ切り替え
Green Light
青信号時の速度へ切り替え

- ④ Computing タブ[Planning] - [Path]のチェックボックスを有効にし、以下を起動します。

[lane_navi]
[lane_rule]
[lane_stop]

- ⑤ Androidタブレットのアプリケーション一覧画面から Autoware Rider を起動します。
- ⑥ [右上メニュー]→[設定]から、以下を設定します。

ROS PC
IP アドレス
ROS PC IPv4 アドレス

命令ポート番号

tablet_receiver ポート番号 (初期値: 5666)

情報ポート番号

tablet_sender ポート番号 (初期値: 5777)

- ⑦ [OK]ボタンをし、ROS PCへ接続を試みます。

このとき設定はファイルに自動的に保存され、次の起動からは保存された設定で接続を試みます。

- ⑧ 画面中央のバーの色が、明るい赤で表示されている場合は接続に成功しています。

表 2 バーの色と接続の状態

バーの色	接続の状態
暗い赤	ROS PC未接続
明るい赤	ROS PC接続
明るい青	自動転 (mode_info: 1)
明るい黄	異常生成 (error_info: 1)

経路データ生成アプリケーションの使用方法

- ① 「AutowareRider」の[NAVI]ボタンを押下し、経路検索を起動します。

- ② 地図を長押しして、以下を順番に実行します。

出発地に設定

目的地に設定

ルート探索実行

- ③ ルート探索の実行後に経路検索を終了することで、ROS PCへ経路データが転送されます。このとき経路データはファイルに自動的に保存され、次回からはルート探索を省略して経路データを転送できます。

- ④ 転送後は、再び「AutowareRider」へ画面が戻ります。

ROS PC への経路データ転送手順

上記の経路データ生成アプリケーションの使用方法 手順③を参照してください(tmp)。

CAN データ収集アプリケーションの使用方法

- ① AutowareRider の[右上メニュー]→[設定]から以下を設定します(tmp)。

データ収集

テーブル名

データ転送先 テーブル名

SSH

ホスト名

SSH 接続先 ホスト名

ポート番号

SSH 接続先 ポート番号 (初期値: 22)

ユーザ名

SSH でログインするユーザ名

パスワード

SSH でログインするパスワード

ポートフォワーディング

ローカルポート番号

ローカルマシンの転送元ポート番号 (初期値: 5558)

リモートホスト名

リモートマシン ホスト名 (初期値: 127.0.0.1)

リモートポート番号

リモートマシンの転送先ポート番号 (初期値: 5555)

- ② [OK]ボタンを押下することで、設定がファイルに保存されます。

ただし、SSH のパスワードはファイルに保存しません。AutowareRider を起動している間だけ、メモリにのみ保持しています。

③ [右上メニュー]→[データ収集]から、以下のいずれかを起動します。

- CanGather
- CarLink (Bluetooth)
- CarLink (USB)

④ アプリケーション起動後の使用方法は、それぞれを単独で起動した場合と同様です。

詳細は、以下の URL を参考にしてください。

<https://github.com/CPFL/Autoware/blob/master/vehicle/general/android/README.md>

ROS PC への CAN データ転送手順

上記の [CANデータ収集アプリケーションの使用方法](#) 手順④を参照してください(tmp)。

Launch ファイルの起動方法

① 「AutowareRider」 の[S1]ボタン、[S2]ボタンは、それぞれが以下の Launch ファイルに対応しています(tmp)。

- check.launch
- set.launch

ボタンを押下することで、ROS PC で Launch ファイルが起動します。

表 3 ボタンと Launch ファイルの状態

ボタン	Launch ファイルの状態
押下 色文字色 黒	起動 ({ndt, lf}_stat: false)
押下 色文字色 赤	起動 ({ndt, lf}_stat: true)

チャプター
4

4. Autoware の画面解説

Runtime Manager

Runtime Manager は runtime_manager パッケージに含まれる Python スクリプト (scripts/runtime_manager_dialog.py) を rosrun コマンドで起動します。

```
$ rosrun runtime_manager runtime_manager_dialog.py
```

Runtime Manager のダイアログの画面は、複数のタブ画面で構成されます。

Runtime Manager のダイアログ操作により、Autoware で使用する各種 ROS ノードの起動・終了処理や、起動した各種 ROS ノードへのパラメータ用のトピックの発行処理などを実行することができます。

各種 ROS ノードを起動／終了するためのボタン類は、ノードの機能により、各タブ画面に分類・配置されています。

各タブ画面は、画面上部のタブを選択することによって表示を切替えます。

Runtime Manager - Quick Start タブ

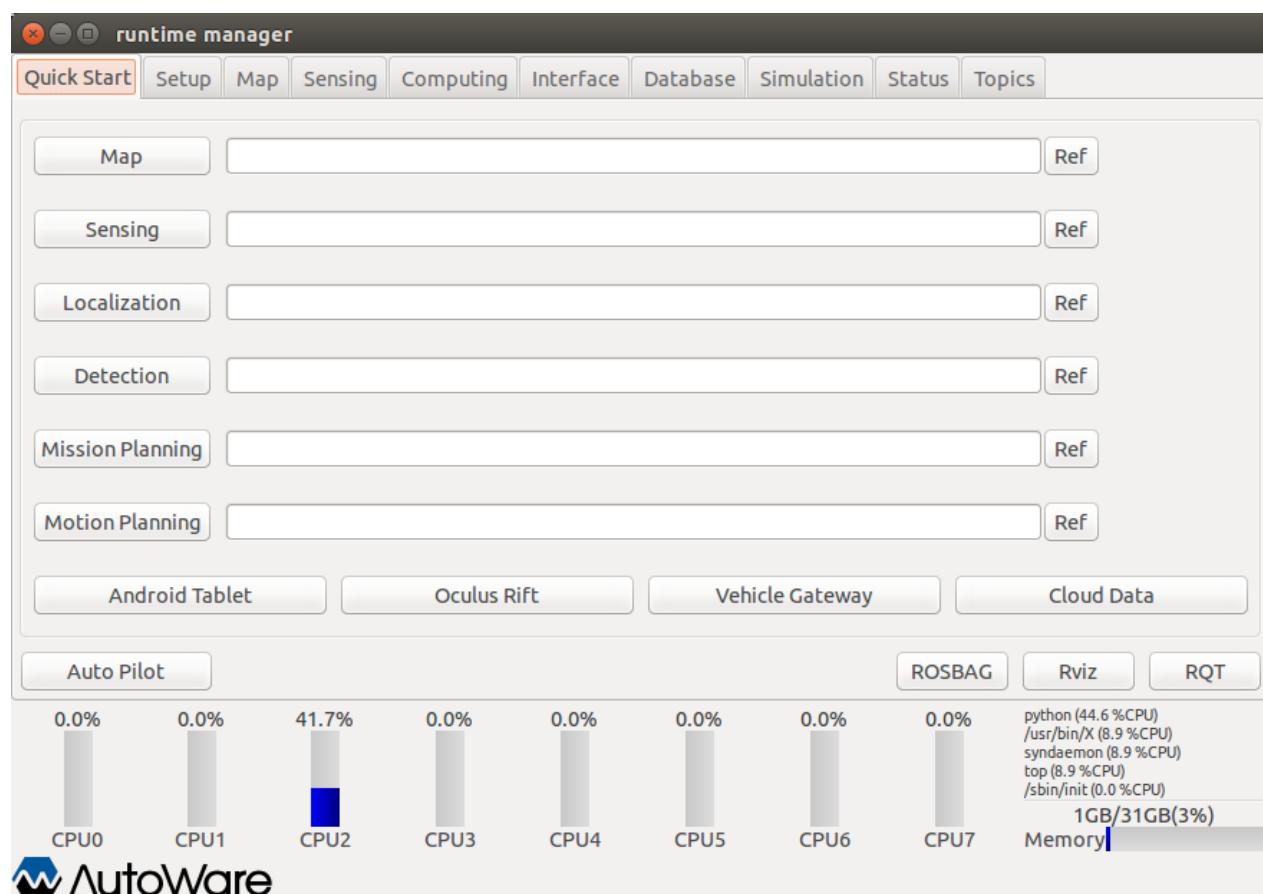


図 17 Runtime Manager - Quick Start タブ

- ❖ [Map] … [Map] テキストボックスでフルパスで指定した .launch スクリプトを起動／終了します。 [Ref] ボタンでファイル選択ダイアログを表示し、スクリプトを選択することもできます。
- ❖ [Sensing] … [Sensing] テキストボックスでフルパスで指定した .launch スクリプトを起動／終了します。 [Ref] ボタンでファイル選択ダイアログを表示し、スクリプトを選択することもできます。
- ❖ [Localization] … [Localization] テキストボックスでフルパスで指定した .launch スクリプトを起動／終了します。 [Ref] ボタンでファイル選択ダイアログを表示し、スクリプトを選択することもできます。
- ❖ [Detection] … [Detection] テキストボックスでフルパスで指定した .launch スクリプトを起動／終了します。 [Ref] ボタンでファイル選択ダイアログを表示し、スクリプトを選択することもできます。

- ❖ [Mission Planning] … [Mission Planning] テキストボックスでフルパスで指定した .launch スクリプトを起動／終了します。[Ref] ボタンでファイル選択ダイアログを表示し、スクリプトを選択することもできます。
- ❖ [Motion Planning] … [Motion Planning] テキストボックスでフルパスで指定した .launch スクリプトを起動／終了します。[Ref] ボタンでファイル選択ダイアログを表示し、スクリプトを選択することもできます。
- ❖ [Android Tablet] ボタン … runtime_manager/tablet_socket.launch スクリプトを起動／終了します。
- ❖ [Oculus Rift] ボタン … <未実装>
- ❖ [Vehicle Gateway] ボタン … runtime_manager/vehicle_socket.launch スクリプトを起動／終了します。
- ❖ [Cloud Data] ボタン … obj_db/obj_downloader ノードを起動／終了します。
- ❖ [Auto Pilot] ボタン … ボタンの状態に応じた mode_cmd トピックを発行します。
- ❖ [ROSBAG] ボタン … ROSBAG Record ダイアログを表示します。
- ❖ [Rviz] ボタン … rviz/rivz ノードを起動／終了します。
- ❖ [RQT] ボタン … rqt を起動／終了します。

Runtime Manager - Setup タブ

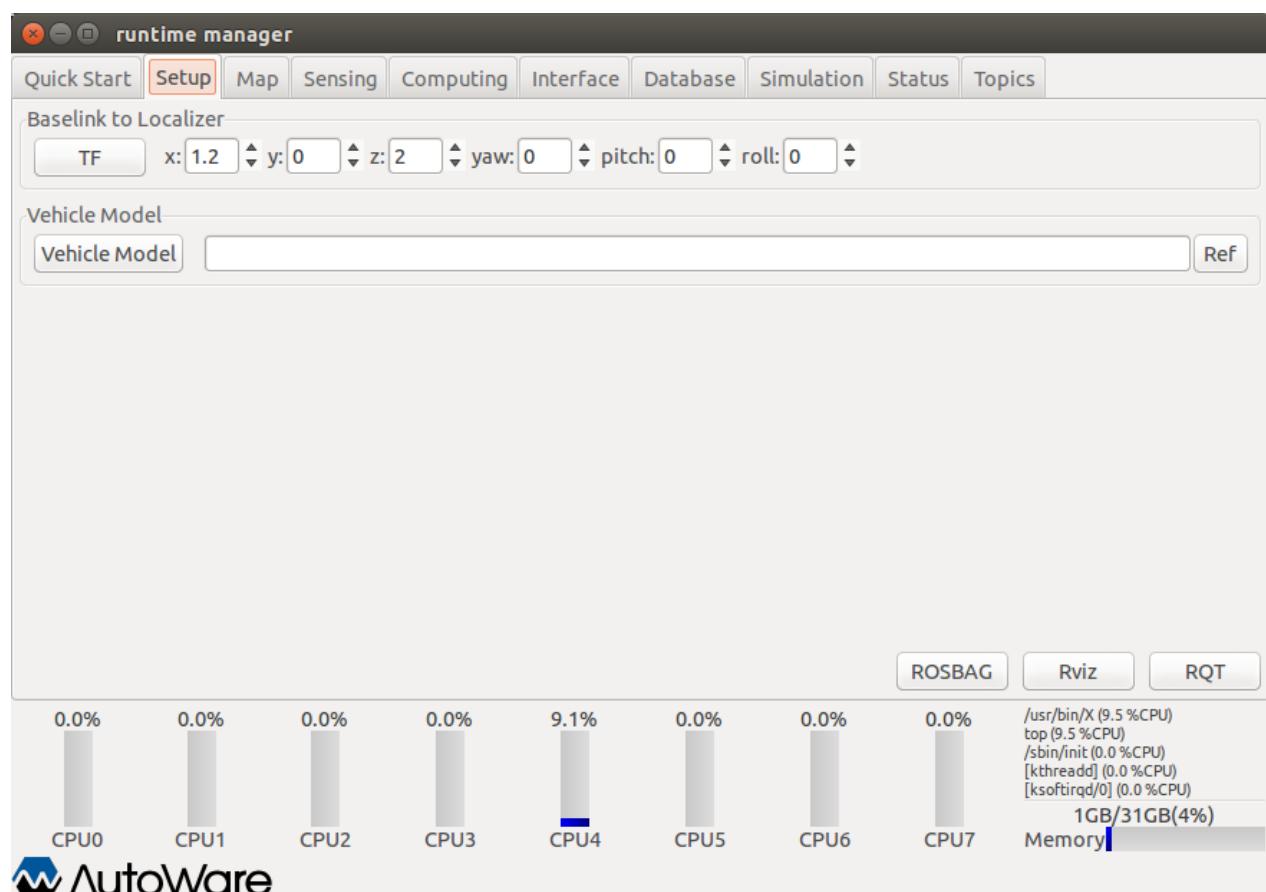


図 18 Runtime Manager - Setup タブ

[Baselink to Localizer]

- ❖ [TF] …base_link → velodyne の tf をパブリッシュします(tmp)。
- ❖ [x], [y], [z], [yaw], [pitch], [roll] …車両の制御位置(base_link)と Velodyne の位置関係を入力します(tmp)。

[Vehicle Model]

- ❖ [Vehicle Model] … [Vehicle Model] テキストボックスでフルパスで指定したファイルを設定します(tmp)。

Runtime Manager - Map タブ

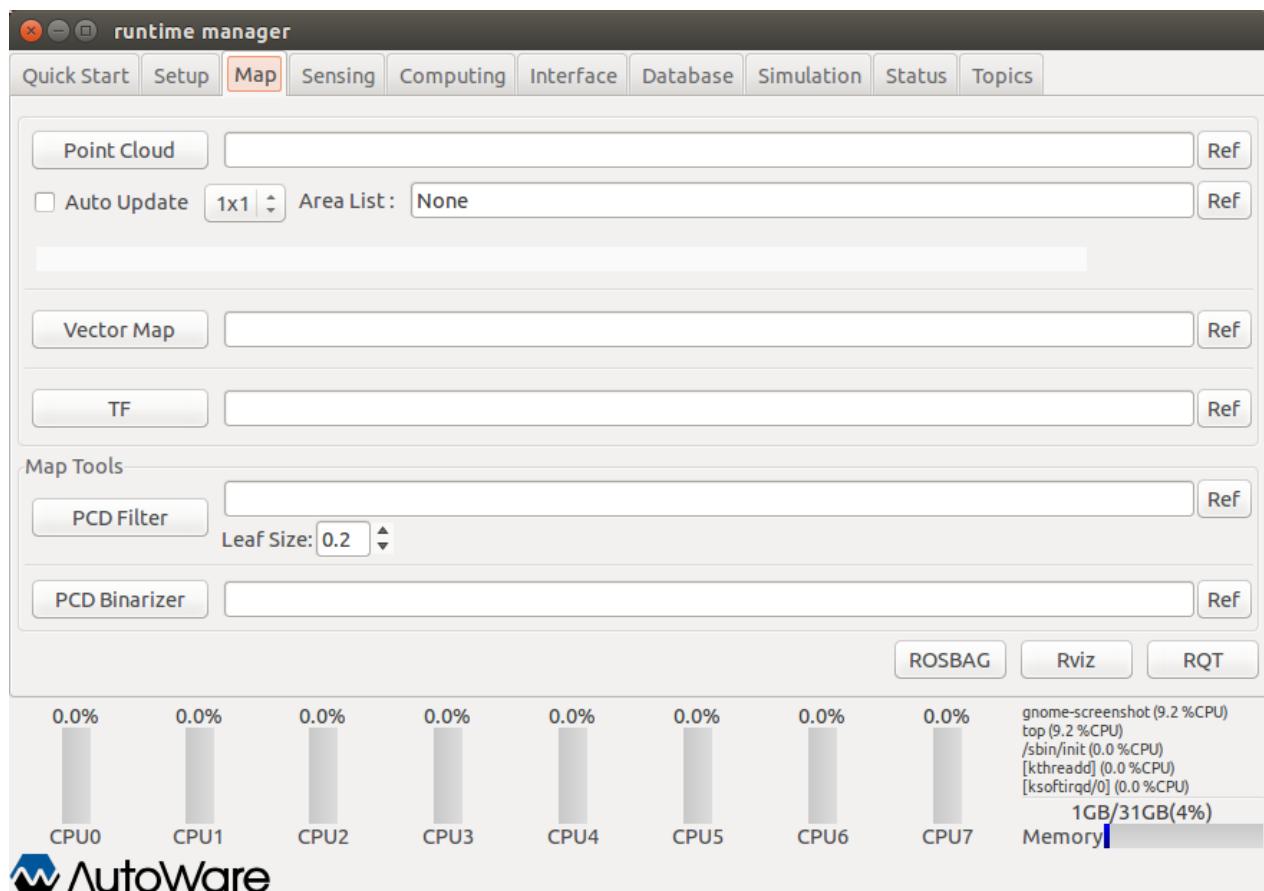


図 19 Runtime Manager - Map タブ

- ❖ [Point Cloud] … [Point Cloud] テキストボックスでフルパスで指定した、pcd ファイル群を引数として map_file/points_map_loader ノードを起動・終了します。[Ref] ボタンでファイル選択ダイアログを表示し、スクリプトを選択することもできます。
- ❖ [Auto Update] … map_file/points_map_loader を起動する際の、自動アップデートの有無を指定します。ドロップダウンボックスでは、自動アップデート有効時のシン数を指定します（Auto Update チェックボックスで ON が指定された場合のみ有効）。Area List テキストボックスで、map_file/points_map_loader を起動する際に引数で渡す area list ファイルのパスをフルパスで指定します。
- ❖ [Vector Map] … [Vector Map] テキストボックスでフルパスで指定した csv ファイル群を引数に、map_file/vector_map_loader ノードを起動・終了します。[Ref] ボタンでファイル選択ダイアログを表示し、スクリプトを選択することもできます。
- ❖ [TF] … [TF] テキストボックスでフルパスで指定した launch ファイルを起動・終了します。TF テキストボックスに launch ファイルが設定されていない場合は、

「`~/.autoware/data/tf/tf.launch`」の launch ファイルを起動・終了します。 [Ref] ボタンでファイル選択ダイアログを表示し、ファイルを選択することもできます。

[Map Tools]

- ❖ [PCD Filter] … [PCD Filter] テキストボックスでフルパスで指定したファイルを設定します(tmp)。
- ❖ [PCD Binarizer] … [PCD Binarizer] テキストボックスでフルパスで指定したファイルを設定します(tmp)。
- ❖ [ROSBAG] ボタン … ROSBAG Record ダイアログを表示します。
- ❖ [Rviz] ボタン … rviz/rivz ノードを起動／終了します。
- ❖ [RQT] ボタン … rqt を起動／終了します。

Runtime Manager - Sensing タブ

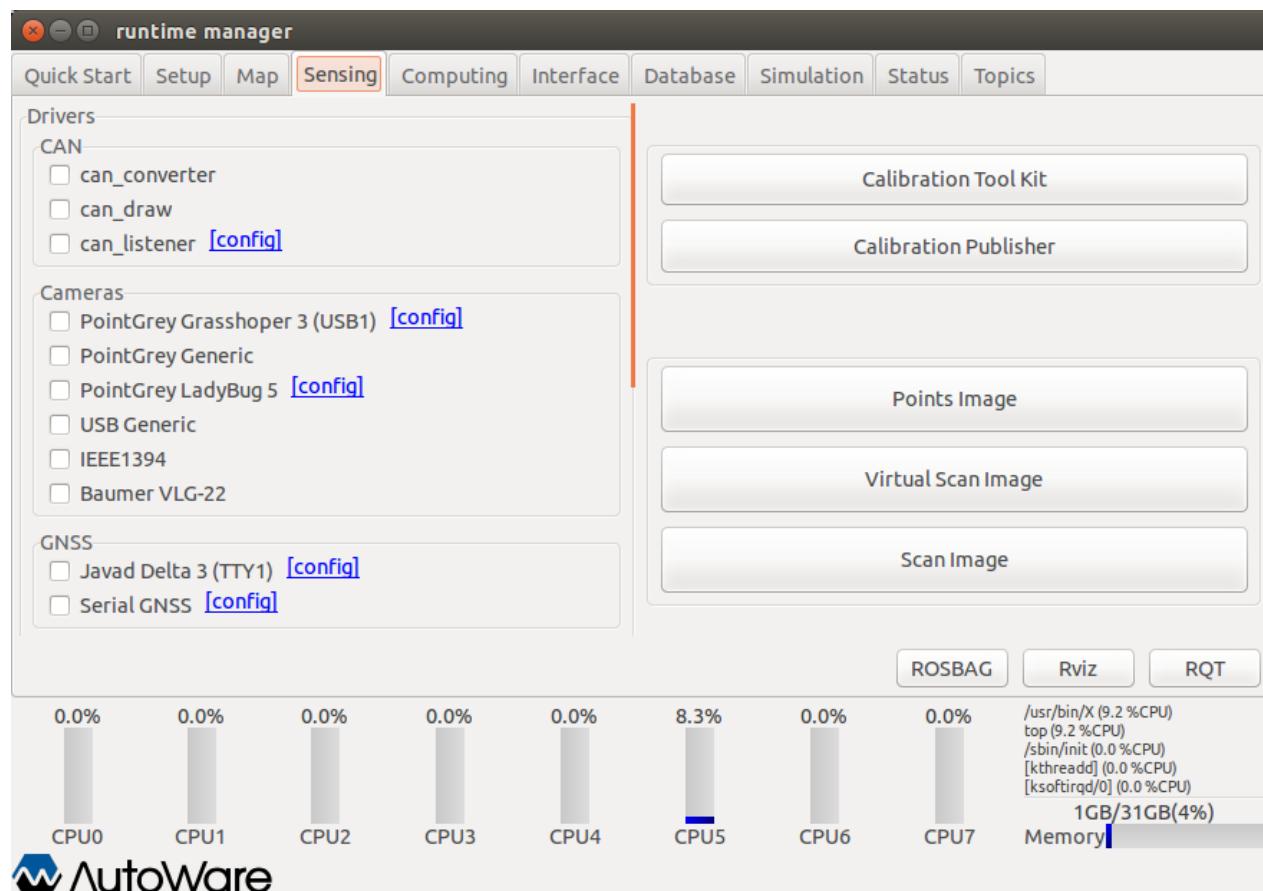


図 20 Runtime Manager - Sensing タブ

[Drivers]

-[CAN]

- ❖ [can_converter] チェックボックス … kvaser/can_converter ノードを起動・終了します。
- ❖ [can_draw] チェックボックス … kvaser/can_draw ノードを起動・終了します。
- ❖ [can_listener] チェックボックス … kvaser/can_listener ノードを起動・終了します。
- ❖ [can_listener]-[config] … can_listener ダイアログを表示します。ノード起動時に指定するチャンネルを設定します。

-[Cameras]

- ❖ [PointGrey Grasshopper 3 (USB1)] チェックボックス …
pointgrey/grasshopper3.launch スクリプトを起動・終了します。
- ❖ [PointGrey Grasshopper 3 (USB1)] -[config] …
calibration_path_grasshopper3 ダイアログを表示します。スクリプト起動時に指定する CalibrationFile の path を設定します。
- ❖ [PointGrey Generic] チェックボックス … ノードを起動・終了します(tmp)。
- ❖ [PointGrey PointGray LadyBug 5] チェックボックス … <未実装>
- ❖ [PointGrey PointGray LadyBug 5] -[config] … ダイアログを表示します(tmp)。
- ❖ [USB Generic] チェックボックス … uvc_camera/uvc_camera_node ノードを起動・終了します。
- ❖ [IEEE1394] チェックボックス … ノードを起動・終了します(tmp)。
- ❖ [Baumer VLG-22] チェックボックス … ノードを起動・終了します(tmp)。

-[GNSS]

- ❖ [Javad Delta 3(TTY1)] チェックボックス … avad/gnss.sh スクリプトを起動・終了します。
- ❖ [Javad Delta 3(TTY1)]-[config] … ダイアログを表示します(tmp)。
- ❖ [Serial GNSS] チェックボックス … ノードを起動・終了します(tmp)。
- ❖ [Serial GNSS]-[config] … ダイアログを表示します(tmp)。

- ❖ [Calibration Tool Kit] ボタン … camera_lidar3d/camera_lidar3d_offline_calib ノードを起動・終了します。
- ❖ [Calibration Publisher] ボタン … calibration_camera_lidar/calibrtion_publisher ノードを起動・終了します。起動時に、calibration_publisher ダイアログを表示するので、ノード起動時に指定する YAML ファイルのパスをフルパスで指定します。

- ❖ [Points Image] ボタン … points2image/points2image ノードを起動・終了します。
- ❖ [Virtual Scan Image] ボタン … runtime_manager/vscan.launch スクリプトを起動・終了します。

- ❖ [Scan Image] ボタン … scan2image/scan2image ノードを起動・終了します。
- ❖ [ROSBAG]ボタン … ROSBAG Record ダイアログを表示します。
- ❖ [Rviz]ボタン … rviz/rivz ノードを起動／終了します。
- ❖ [RQT]ボタン … rqt を起動／終了します。

Runtime Manager - Computing タブ

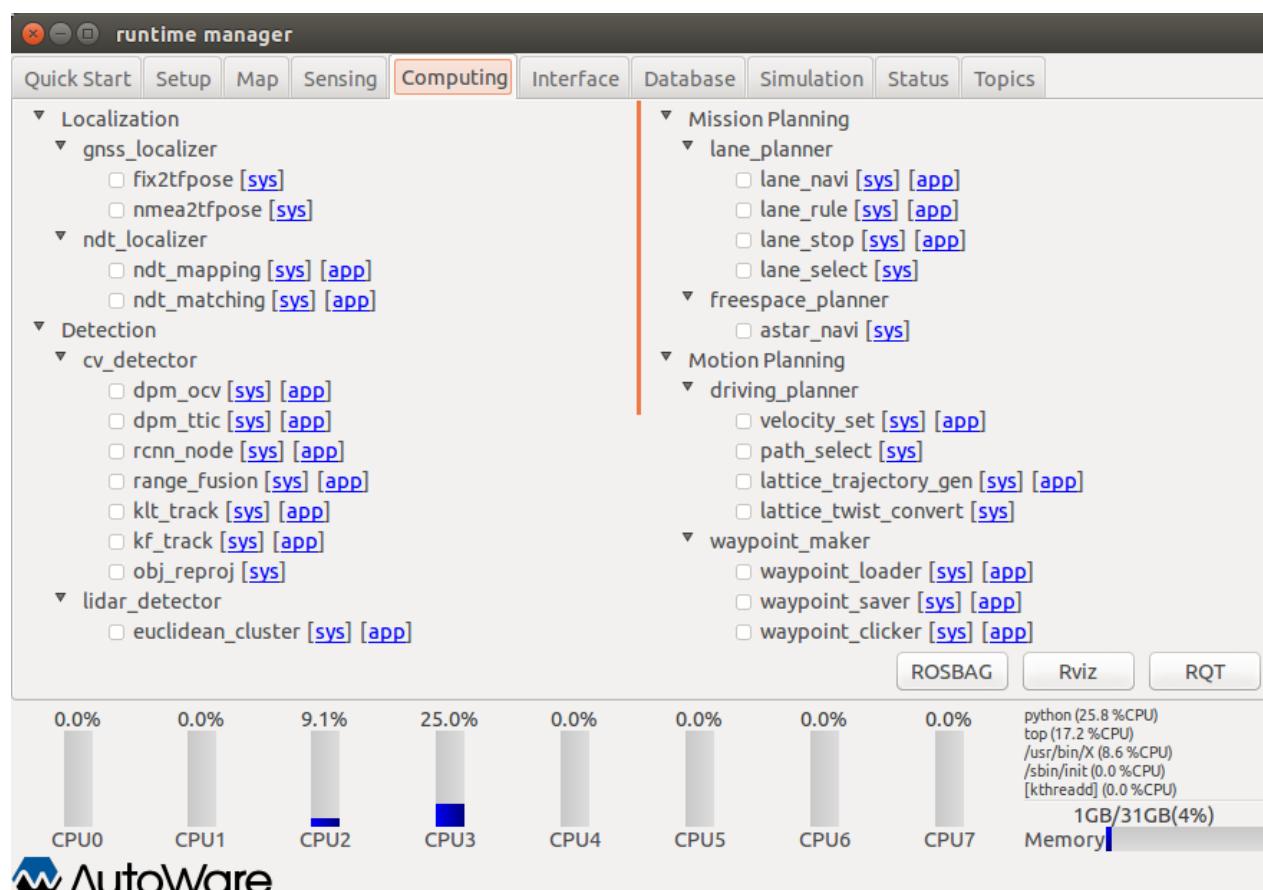


図 21 Runtime Manager - Computing タブ

[Localization]

-[gnss_lozalizer]

- ❖ [fix2tfpose] チェックボックス … gnss_localizer/fix2tfpose ノードを起動・終了します。
- ❖ [nmea2tfpose] チェックボックス … gnss_localizer/nmea2tfpose.launch スクリプトを起動・終了します。

-[ndt_lozalizer]

- ❖ [ndt_mapping] チェックボックス … ndt_localizer/ndt_mapping.launch スクリプトを起動・終了します。[app]をクリックすると「ndt_mapping」ダイアログを表示します(tmp)。

- ❖ [ndt_matching] チェックボックス … ndt_localizer/ndt_matching.launch スクリプトを起動・終了します。[app]をクリックすると「ndt」ダイアログを表示します(tmp)。

[Detection]

-[cv_detector]

- ❖ [dpm_ocv] チェックボックス … runtime_manager/dpm_ocv.launch スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します。ダイアログではチューニングするパラメータの種別(Car 又は Pedestrian)を選択すると、car_dpm ダイアログ又は pedestrian_dpm ダイアログを表示します。パラメータ変更、/config/car_dpm 又は/config/pedestrian_dpm トピックを発行します(tmp)。
- ❖ [dpm_ocv] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [dpm_ttic] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [rcnn_node] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [range_fusion] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [klt_track] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [kf_track] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [obj_reproj] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。

[lidar_detector]

- ❖ [euclidean_cluster] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します(tmp)。

[Mission Planning]

-[lane_planner]

- ❖ [lane_navi] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [lane_rule] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [lane_stop] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [lane_select] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [lane_navi] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。

-[freespace_planner]

- ❖ [astar_navi] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。

[Motion Planning]

-[driving_planner]

- ❖ [velocity_set] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [path_select] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。

- ❖ [lattice_trajectory_gen] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [lattice_twice_convert] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。

-[waypoint_marker]

- ❖ [waypoint_loader] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [waypoint_saver] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [waypoint_clicker] チェックボックス …スクリプトを起動・終了します。起動時に dpm_ocv ダイアログが表示されます。パラメータを設定後、Detection Start ボタンでスクリプトが起動します。[app]をクリックするとダイアログを表示します (tmp)。
- ❖ [ROSBAG] ボタン … ROSBAG Record ダイアログを表示します。
- ❖ [Rviz] ボタン … rviz/rivz ノードを起動／終了します。
- ❖ [RQT] ボタン … rqt を起動／終了します。

Runtime Manager - Interface タブ

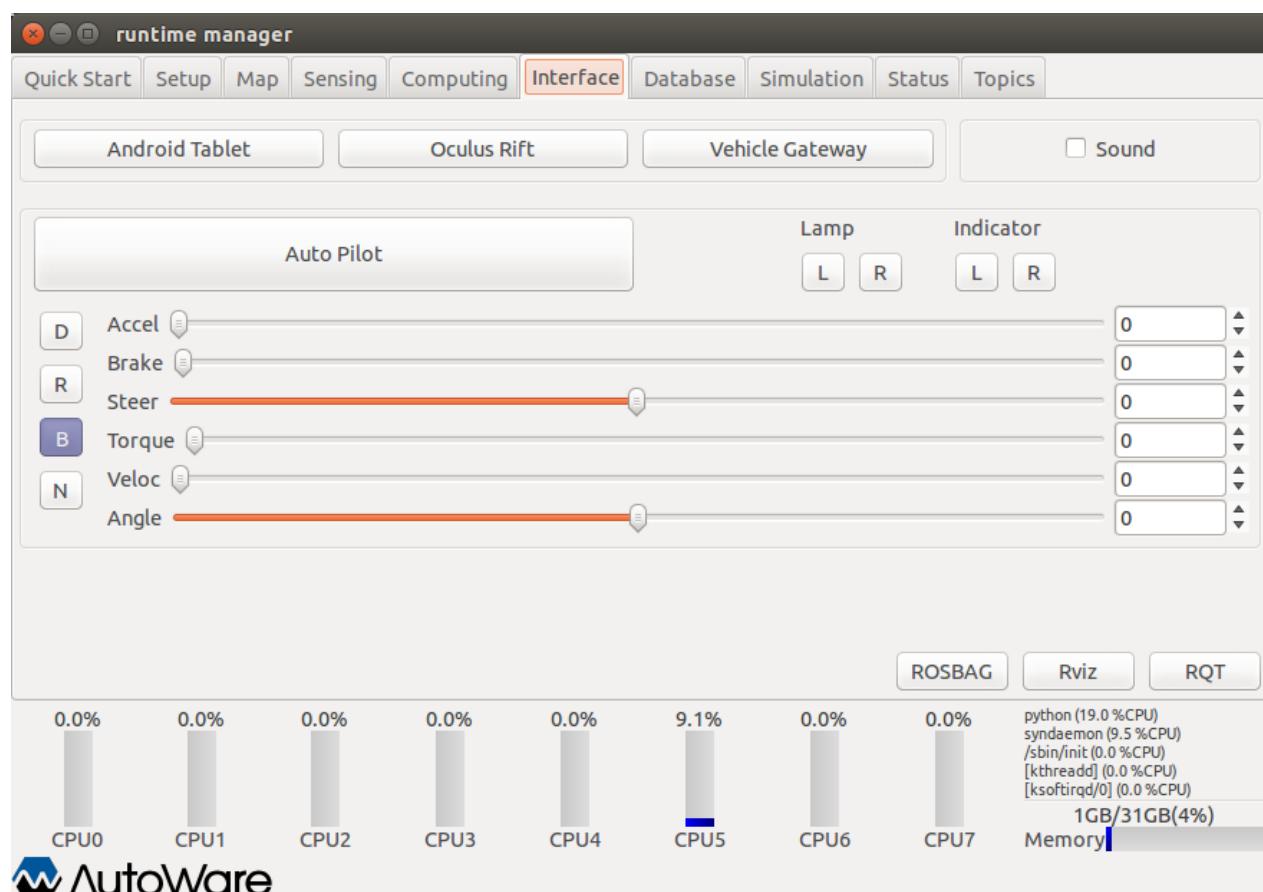


図 22 Runtime Manger - Interface タブ

- ❖ [Android Tablet] ボタン … runtime_manager/tablet_socket.launch スクリプトを起動・終了します。
- ❖ [Oculus Rift] ボタン … <未実装>
- ❖ [Vehicle Gateway] ボタン … runtime_manager/vehicle_socket.launch スクリプトを起動・終了します。
- ❖ [Sound] チェックボックス … sound_player/sound_player.py スクリプトを起動・終了します。
- ❖ [Auto Pilot] ボタン … ボタンの状態に応じた mode_cmd トピックを発行します。
- ❖ [Lamp] … ボタンの状態に応じた lamp_cmd トピックを発行します。
- ❖ [Indicator] … ボタンの状態に応じた indicator_cmd トピックを発行します。

- ❖ [D] トグルスイッチ … ON 操作したボタンに応じた gear_cmd トピックを発行します。
- ❖ [R] トグルスイッチ … ON 操作したボタンに応じた gear_cmd トピックを発行します。
- ❖ [B] トグルスイッチ … ON 操作したボタンに応じた gear_cmd トピックを発行します。
- ❖ [N] トグルスイッチ … ON 操作したボタンに応じた gear_cmd トピックを発行します。

- ❖ [ROSBAG] ボタン … ROSBAG Record ダイアログを表示します。
- ❖ [Rviz] ボタン … rviz/rivz ノードを起動／終了します。
- ❖ [RQT] ボタン … rqt を起動／終了します。

Runtime Manager - Database タブ

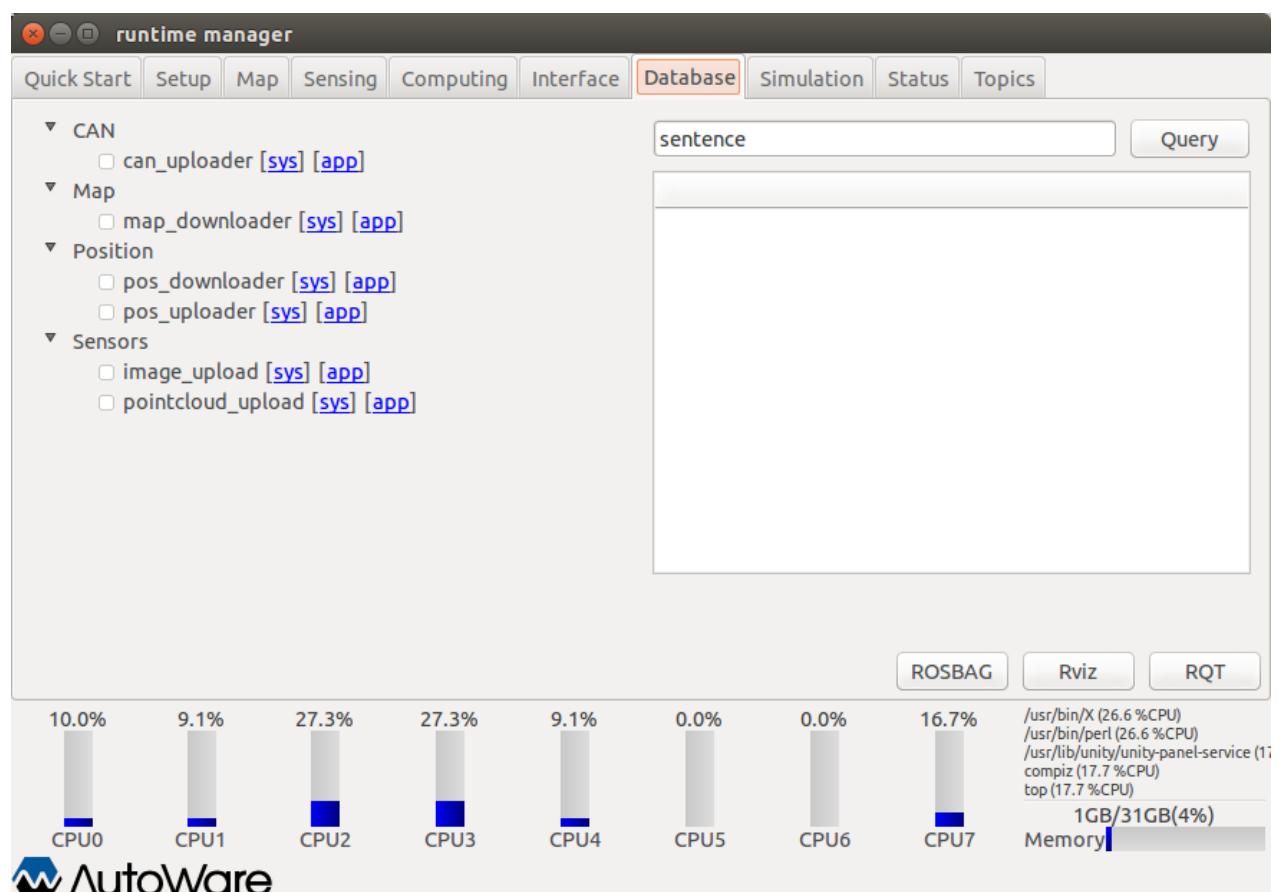


図 23 Runtime Manager - Database タブ

[CAN]

- ❖ [can_uploader] チェックボックス … obj_db/can_uploader ノードを起動・終了します。[app]をクリックすると「other」ダイアログを表示します。

[Map]

- ❖ [map_downloader] チェックボックス … <未実装>。[app]をクリックすると「map_file」ダイアログを表示します。

[Position]

- ❖ [pos_downloader] チェックボックス … pos_db/pos_downloader ノードを起動・終了します。[app]をクリックすると「pos_db」ダイアログを表示します。
- ❖ [pos_uploader] チェックボックス … pos_db/pos_uploader ノードを起動・終了します。[app]をクリックすると「pos_db」ダイアログを表示します。

[Sensors]

- ❖ [image_upload] チェックボックス … <未実装>。[app]をクリックすると「other」ダイアログを表示します。
- ❖ [pointcloud_upload] チェックボックス … <未実装>。[app]をクリックすると「other」ダイアログを表示します。
- ❖ [Query]ボタン … <未実装>。
- ❖ [ROSBAG]ボタン … ROSBAG Record ダイアログを表示します。
- ❖ [Rviz]ボタン … rviz/rivz ノードを起動／終了します。
- ❖ [RQT]ボタン … rqt を起動／終了します。

Runtime Manager - Simulation タブ

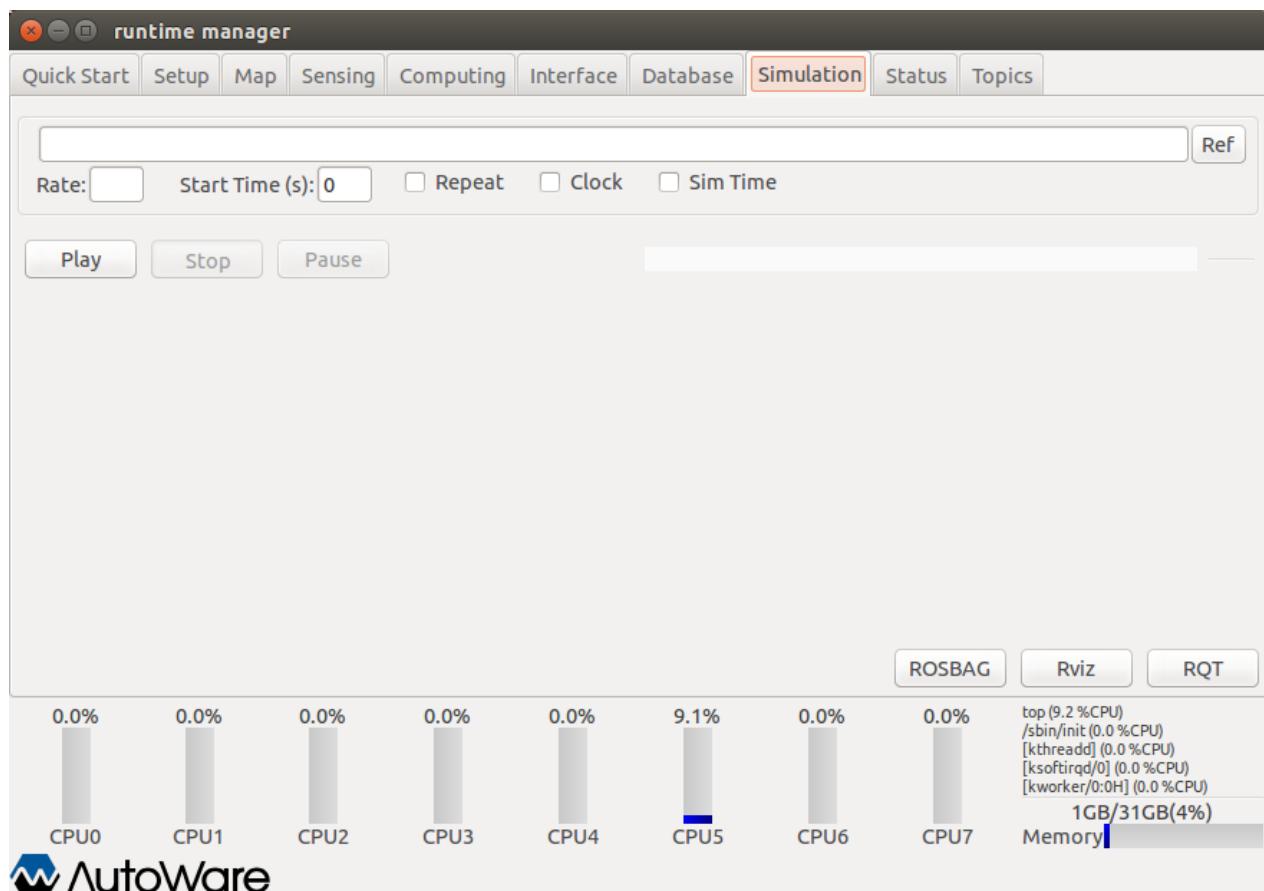


図 24 Runtime Manager - Simulation タブ

- ❖ [ROSBAG] テキストボックス … テキストボックスでフルパスで指定した、 bag ファイルを引数として rosbag play を実行します。 [Ref] ボタンでファイル選択ダイアログを表示し、 bag ファイルを選択することもできます。
- ❖ [Rate] … rosbag play コマンドを起動する際の -r オプションで指定する数値を指定します。未設定の場合は -r オプションを指定しません。
- ❖ [State Time(s)] … rosbag play コマンドを起動する際の -start オプションで指定する数値を指定します。未設定の場合は -start オプションを指定しません。
- ❖ [Repeat] チェックボックス … ON の場合、 rosbag play コマンドを起動する際に、 --loop オプションが指定されます。
- ❖ [Clock] チェックボックス … ON の場合、 rosbag play コマンドを起動する際に、 --clock オプションが指定されます。この設定は終了時に保存されません。

- ❖ [Sim Time] チェックボックス …rosparam /use_sim_time の設定値 (true/false) を表示します。チェックボックスを操作すると、値を rosparam /usr_sim_time に設定します。この設定は終了時に保存されません。
- ❖ [Play] ボタン …ROSBAG テキストボックスに設定された bag ファイルを指定して、rosbag play コマンドを起動します。
- ❖ [Stop] ボタン …起動している rosbag play コマンドを終了します。
- ❖ [Pause] ボタン …起動している rosbag play コマンドを一時停止します。

- ❖ [ROSBAG] ボタン … ROSBAG Record ダイアログを表示します。
- ❖ [Rviz] ボタン … rviz/rivz ノードを起動／終了します。
- ❖ [RQT] ボタン … rqt を起動／終了します。

Runtime Manager - Status タブ

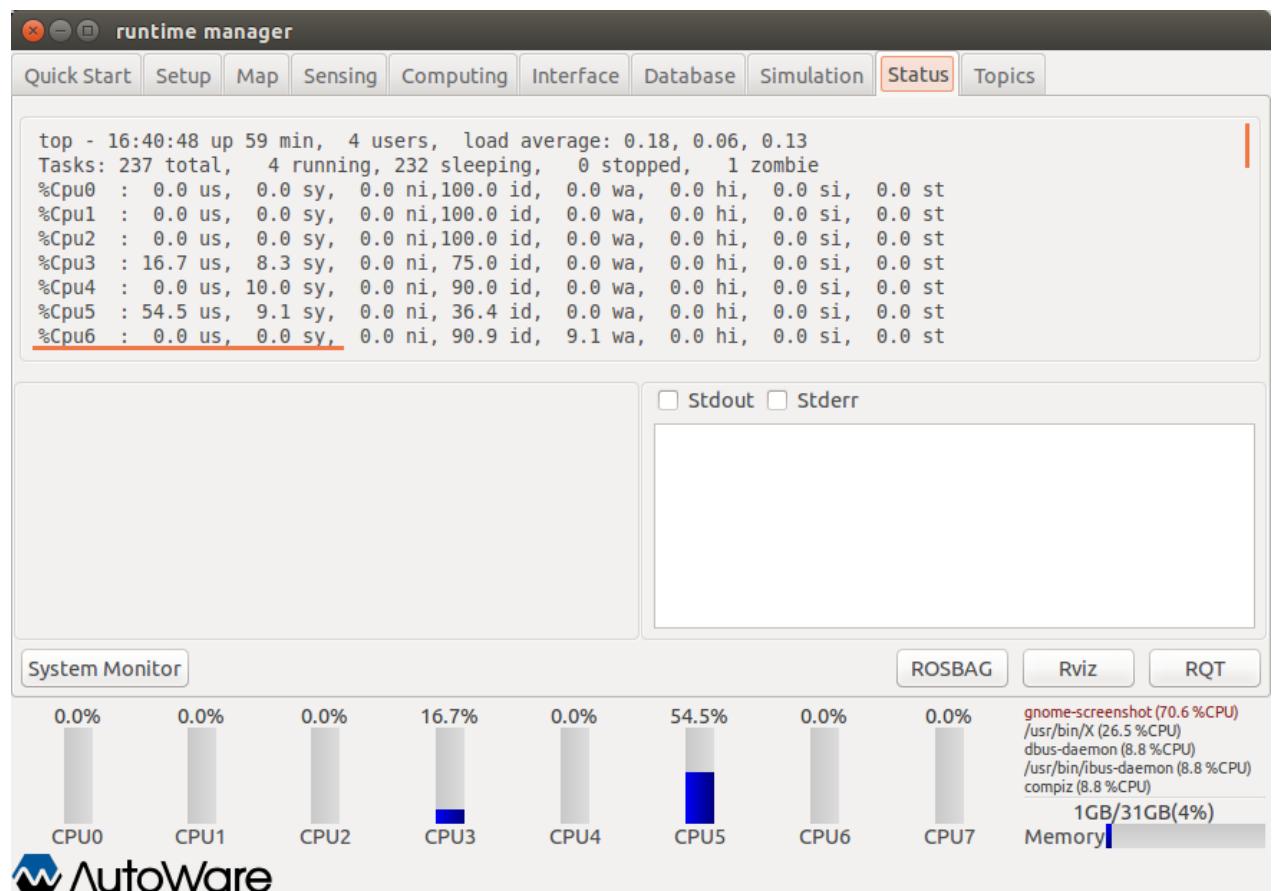


図 25 Runtime Manager - Status タブ

- ❖ 上段の表示 … 内部で実行している top コマンドの実行結果を表示します。
 - ❖ 下段左側の表示 … 関連のノードが発行する周期実行時間を表示します。
 - ❖ 下段右側の表示 … 起動したノード、スクリプトの標準出力、標準エラー出力の内容を表示します。ただし、プログレスバー表示を行う一部のノードでは表示されません。
 - ❖ 最下部の情報表示 … CPU(コア)の負荷状況とメモリ使用量を表示します。
 - ❖ [System Monitor] ボタン … (記載なし(tmp))
 - ❖ [ROSBAG] ボタン … ROSBAG Record ダイアログを表示します。
 - ❖ [Rviz] ボタン … rviz/rivz ノードを起動／終了します。

❖ [RQT]ボタン … rqt を起動／終了します。

Runtime Manager - Topics タブ

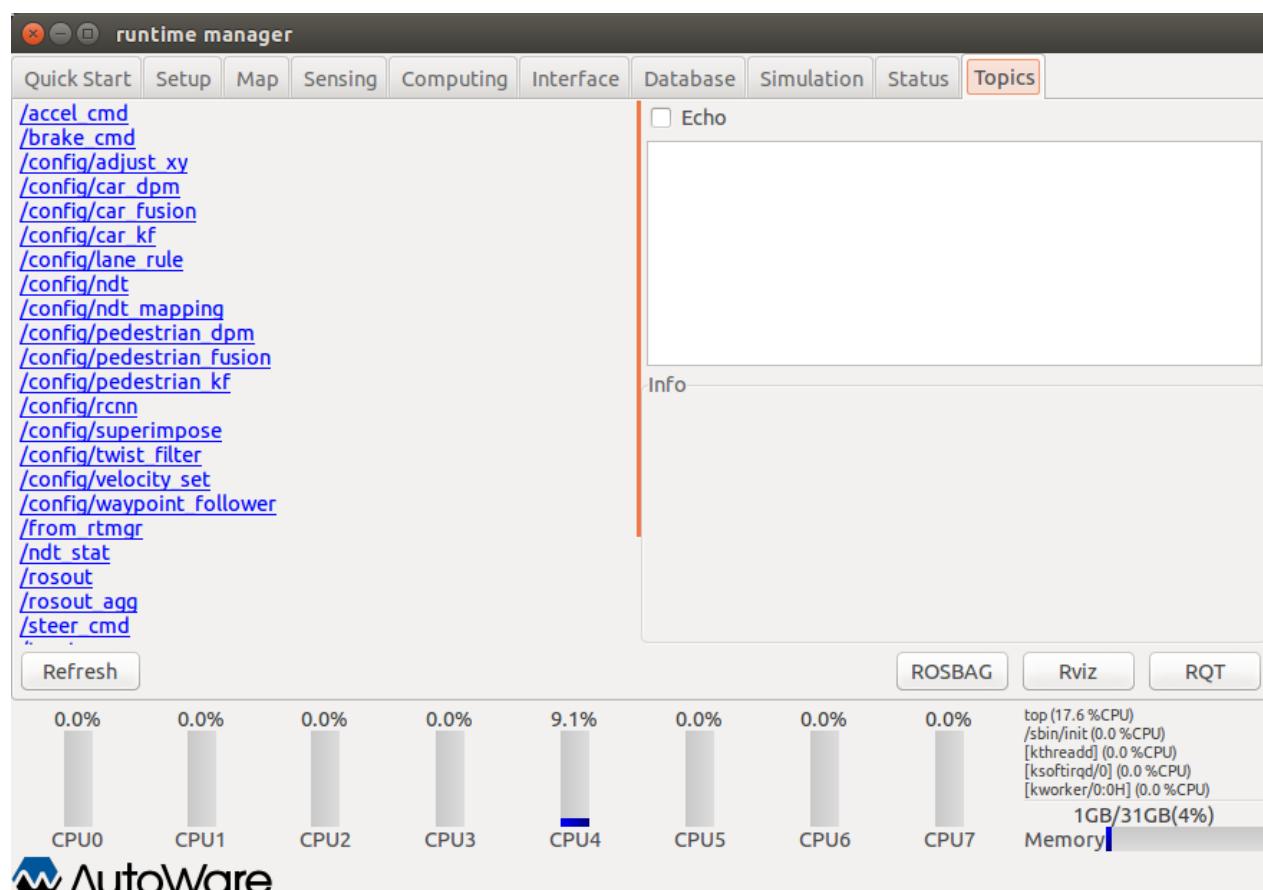


図 26 Runtime Manager - Topics タブ

- ❖ 左側の表示 … トピック名の一覧を表示します。リンクをクリックすると、`rostopic echo <対象トピック>` コマンドを実行し、右側上段に結果を表示し、`rostopic info <対象トピック>` コマンドを実行し、右側下段に結果を表示します。(tmp)
- ❖ [Refresh]ボタン … (記載なし(tmp))
- ❖ [ROSBAG]ボタン … ROSBAG Record ダイアログを表示します。
- ❖ [Rviz]ボタン … rviz/rivz ノードを起動／終了します。
- ❖ [RQT]ボタン … rqt を起動／終了します。

ROSBAG Record ダイアログ

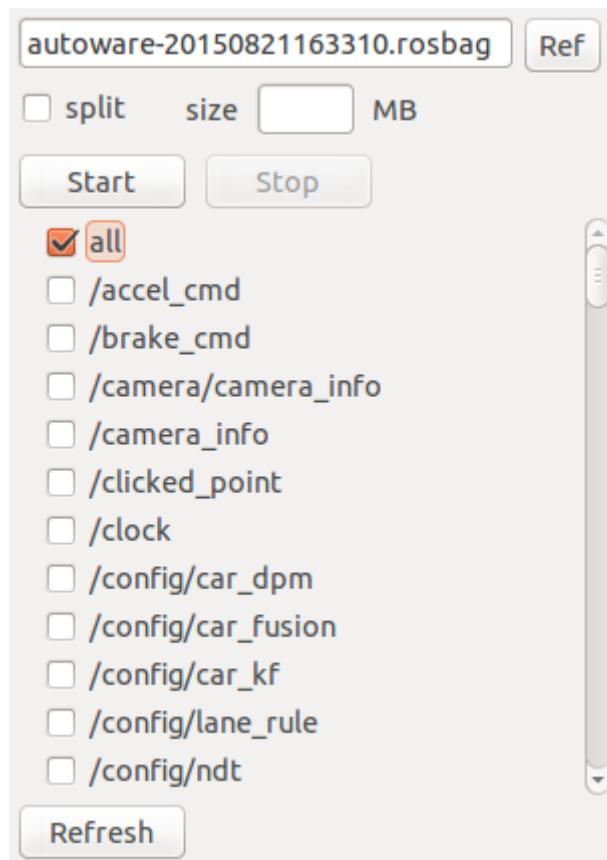


図 27 ROSBAG Record ダイアログ

- ❖ **テキストボックス**…rosbag record コマンドを実行する際の、 bag ファイルをフルパスで指定します。[Ref]ボタンでファイル選択ダイアログを表示し、スクリプトを選択することもできます。
- ❖ **[Start]ボタン**… 上部テキストボックスに設定された bag ファイルを指定して、 rosbag record コマンドを起動します。
- ❖ **[Stop]ボタン**… 起動している rosbag record コマンドを終了します。
- ❖ **[All] チェックボックス**…チェックボックスが ON の場合、 rosbag record コマンドを起動する際に、 -a オプションが指定されます。
- ❖ **(その他チェックボックス群)** … rosbag record コマンドを起動する際に、 チェックボックスが ON のトピックを指定します。但し、 All チェックボックスが OFF の場合のみ有効です。

- ❖ [Refresh]ボタン … rostopic list コマンドを実行し、現在有効なトピックを調べて、その他のチェックボックス群を更新します。
- ❖ (最下行の情報表示) … CPU(コア)の負荷状況とメモリ使用量を表示します。

Rviz

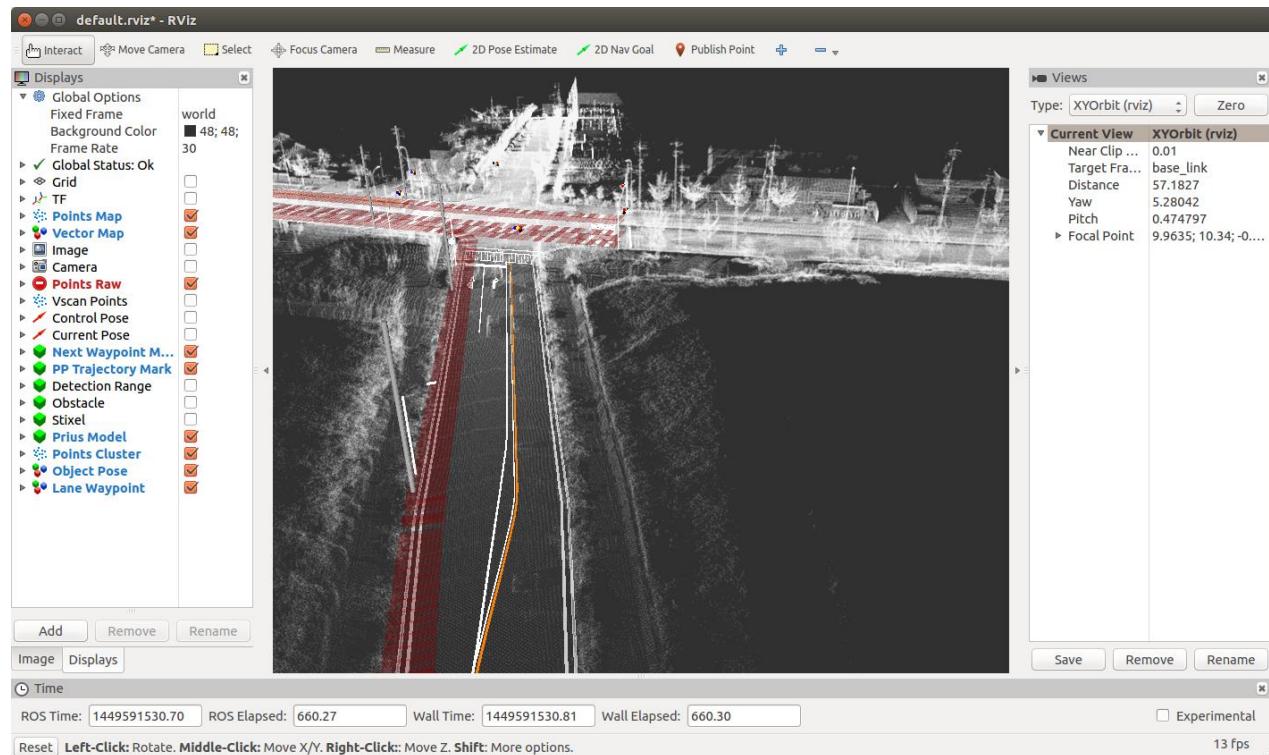


图 28 Rviz

AutowareRider

Knight Rider に似た UI を持った、Android アプリケーションです。

以下が起動時の画面です。

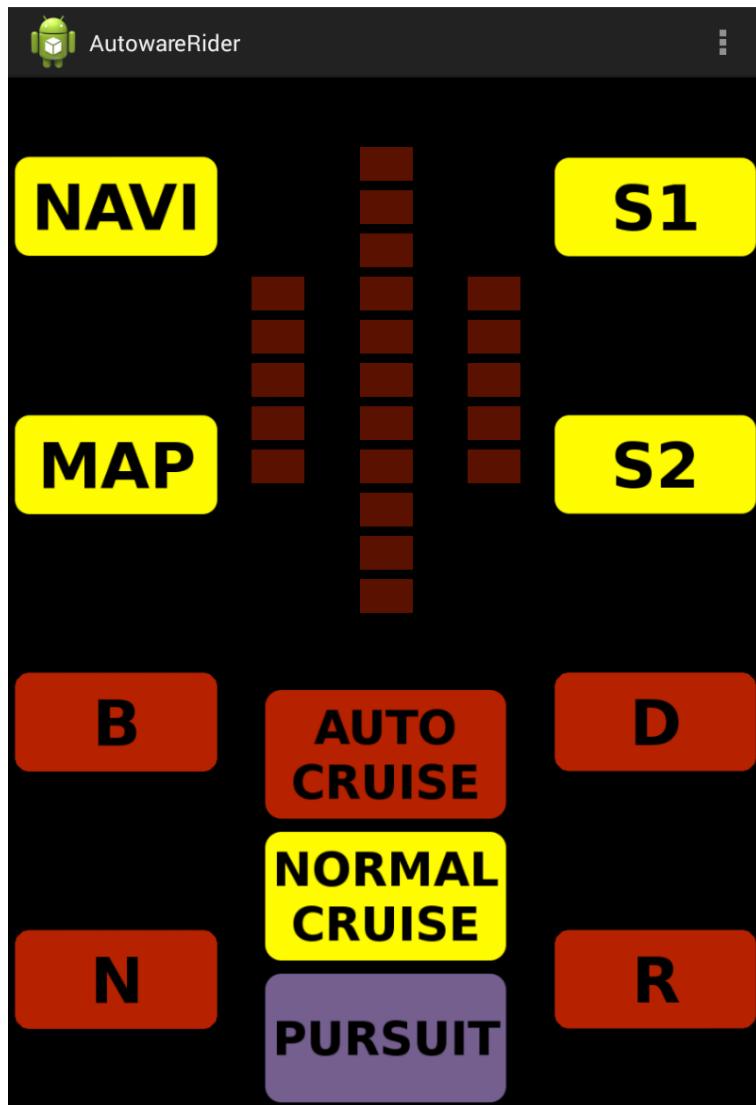


図 29 AutowareRider

- ❖ [Navi]ボタン … AutowareRoute.apk を起動します。
- ❖ [MAP]ボタン … <未実装>
- ❖ [S1]ボタン … check.launch を ROS PC で起動します。
- ❖ [S2]ボタン … set.launch を ROS PC で起動します。

- ❖ [B]ボタン … ギア情報 B を ROS PC へ送信します。
- ❖ [N]ボタン … ギア情報 N を ROS PC へ送信します。
- ❖ [D]ボタン … ギア情報 D を ROS PC へ送信します。
- ❖ [R]ボタン … ギア情報 R を ROS PC へ送信します。
- ❖ [AUTO CRUISE]ボタン … <未実装>
- ❖ [NORMAL CRUISE]ボタン … <未実装>
- ❖ [PURSUIT]ボタン … <未実装>(現状はアプリケーションを終了します)

[右上メニュー]から以下が選択できます。

[設定]

[データ収集]

以下は[設定]の画面です。



図 30 AutowareRider 設定画面

[ROS PC]

- ❖ IP アドレス … ROS PC IPv4 アドレス
- ❖ 命令受信ポート番号 … tablet_receiver ポート番号 (初期値: 5666)
- ❖ 情報送信ポート番号 … tablet_sender ポート番号 (初期値: 5777)

[データ収集]

- ❖ テーブル名 … データ転送先 テーブル名

[SSH]

- ❖ ホスト名 … SSH 接続先 ホスト名
- ❖ ポート番号 … SSH 接続先 ポート番号 (初期値: 22)
- ❖ ユーザ名 … SSH でログインするユーザ名
- ❖ パスワード … SSH でログインするパスワード

[ポートフォワーディング]

- ❖ ローカルポート番号 … ローカルマシンの転送元ポート番号 (初期値: 5558)
- ❖ リモートホスト名 … リモートマシン ホスト名 (初期値: 127.0.0.1)
- ❖ リモートポート番号 … リモートマシンの転送先ポート番号 (初期値: 5555)

以下は[データ収集]の画面です。



図 31 AutowareRider データ収集画面

- ❖ [CanGather]ボタン …CanGather.apk を起動します。
- ❖ [CarLink (Bluetooth)]ボタン …CarLink_CAN-BT_LS.apk を起動します。
- ❖ [CarLink (USB)]ボタン …CarLink_CANusbAccessory_LS.apk を起動します。

AutowareRoute

AutowareRoute は、MapFan SDK で実装された、経路データ生成のための Android アプリケーションです。

以下が起動時の画面です。



図 32 AutowareRoute

地図を長押しすることで、以下のダイアログが表示されます。



図 33 AutowareRoute ルート設定画面

- ❖ [出発地に設定]ボタン … 長押しした地点を経路データの出発地として設定します。
- ❖ [立寄地に設定]ボタン … 長押しした地点を経路データの立寄地として設定します。
- ❖ [目的地に設定]ボタン … 長押しした地点を経路データの目的地として設定します。
- ❖ [ルート消去]ボタン … ルート探索実行によって生成された経路データを消去します。
- ❖ [ルート探索実行]ボタン … 出発地、立寄地、目的地に応じた経路データを生成します。
- ❖ [終了]ボタン … (記載なし(tmp))

5. 環境構築

対応している OS や必要なソフトウェアについて説明します。

インストール

P C に以下の手順で OS(Linux)、ROS、Autowareなどをインストールします。

OS

2015年9月時点で Autoware が対応している Linux ディストリビューションは以下の通りです。

Ubuntu 13.04

Ubuntu 13.10

Ubuntu 14.04

インストールメディアおよびインストール手順については、以下のサイトを参考にしてください。

Ubuntu Japanese Team

<https://www.ubuntulinux.jp/>

Ubuntu

<http://www.ubuntu.com/>

ROS

Ubuntu14.04 の場合は、下記の手順で ROS および必要なパッケージをインストールします。

```
$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu trusty main" >
\ /etc/apt/sources.list.d/ros-latest.list'
$ wget http://packages.ros.org/ros.key -O - | sudo apt-key add -
$ sudo apt-get update
$ sudo apt-get install ros-indigo-desktop-full ros-indigo-nmea-msgs \ ros-indigo-sound-play
$ sudo apt-get install libnlopt-dev freeglut3-dev qtbase5-dev libqt5opengl5-dev
\ libssh2-1-dev libarmadillo-dev
~/.bashrc などに以下を追加します。
[ -f /opt/ros/indigo/setup.bash ] && . /opt/ros/indigo/setup.bash
```

Ubuntu13.10 もしくは 13.04 の場合は、下記の手順で ROS および必要なパッケージをインストールします。

```
$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" >
\ /etc/apt/sources.list.d/ros-latest.list'
$ sudo apt-get install ros-hydro-desktop-full ros-indigo-nmea-msgs \ ros-hydro-sound-play
$ sudo apt-get install libnlopt-dev freeglut3-dev libssh2-1-dev libarmadillo-dev
~/.bashrc などに以下を追加します。
[ -f /opt/ros/indigo/setup.bash ] && . /opt/ros/indigo/setup.bash
```

Velodyne ドライバ

<https://github.com/ros-drivers/velodyne> からソースコードを入手し、以下の手順でインストールを行います。

```
$ sudo apt-get install libpcap-dev git  
$ mkdir -p ~/ros_drivers/src  
$ cd ~/ros_drivers/src  
$ catkin_init_workspace  
$ git clone https://github.com/ros-drivers/velodyne.git  
$ cd ~/ros_drivers  
$ catkin_make  
$ source devel/setup.bash
```

OpenCV

OpenCV のサイト(<http://sourceforge.net/projects/opencvlibrary/>)から、バージョン 2.4.8 以降のソースコードを入手し、以下の手順でインストールを行います。

```
$ unzip opencv-2.4.8.zip  
$ cd opencv-2.4.8  
$ cmake .  
$ make  
$ sudo make install
```

Qt (必要な場合)

Ubuntu14.04 の場合は qtbase5-dev および libqt5opengl5-dev パッケージはインストール済のため、下記の作業は必要ありません。

- ② Qt5 に必要なパッケージを、以下の手順でインストールします。

```
$ sudo apt-get build-dep qt5-default
$ sudo apt-get install build-essential perl python git
$ sudo apt-get install "libxcb.*" libx11-xcb-dev libglu1-mesa-dev libxrender-dev libxi-dev
$ sudo apt-get install flex bison gperf libicu-dev libxslt-dev ruby
$ sudo apt-get install libssl-dev libxcursor-dev libcomposite-dev libxdamage-dev
\ libxrandr-dev libfontconfig1-dev
$ sudo apt-get install libasound2-dev libgstreamer0.10-dev
\ libgstreamer-plugins-base0.10-dev
```

- ③ 次に、Qt5 のソースコード入手してビルドおよびインストールを行います。

```
$ git clone git://code.qt.io/qt/qt5.git
$ cd qt5/
$ git checkout v5.2.1
$ perl init-repository --no-webkit
    (webkit は大きいため、--no-webkit を指定しています)
$ ./configure -developer-build -opensource -nomake examples -nomake tests
    (ライセンスを受諾する必要があります)
$ make -j
    (ビルドには数時間かかります)
$ make install
$ sudo cp -r qtbase /usr/local/qtbase5
```

CUDA（必要な場合）

NVIDIA 社のグラフィックボードに搭載された GPU を使って計算を行う場合に CUDA が必要となります。

インストールする場合は <http://docs.nvidia.com/cuda/cuda-getting-started-guide-for-linux/> を参考に、以下の手順でインストールします。

① 環境の確認

```
$ lspci | grep -i nvidia
```

(NVIDIA のボードの情報が出力されることを確認)

```
$ uname -m
```

(x86_64 であることを確認)

```
$ gcc --version
```

(インストールされていることを確認)

② CUDA のインストール

<http://developer.nvidia.com/cuda-downloads> から CUDA をダウンロード

(以下、cuda-repo-ubuntu1404_7.0-28_amd64.deb と想定)

```
$ sudo dpkg -i cuda-repo-ubuntu1404_7.0-28_amd64.deb
```

```
$ sudo apt-get update
```

```
$ sudo apt-get install cuda
```

③ システムを再起動（…は不要かもしれません）

```
$ lsmod | grep nouveau
```

(nouveau ドライバがロードされていないことを確認)

④ 確認

```
$ cat /proc/driver/nvidia/version
```

(カーネルモジュール、gcc のバージョンが表示される)

```
$ cuda-install-samples-7.0.sh ~
```

```
$ cd ~/NVIDIA_CUDA-7.0_Samples/1_Utilsilities/deviceQuery/
```

```
$ make
```

```
$ ./deviceQuery
```

⑤ CUDA を普段から使う場合は、以下の設定を .bashrc などに書く

```
export PATH="/usr/local/cuda:$PATH"
```

```
export LD_LIBRARY_PATH="/usr/local/cuda/lib:$LD_LIBRARY_PATH"
```

FlyCapture (必要な場合)

PointGray 社のカメラを使用する場合は、以下の手順で FlyCapture SDK をインストールします。

以下は 2014 年 10 月 28 日に試したときの手順です。

/radisk2/work/usuda/autoware/doc/MultiCameraEclipse-log-20141028.txt

① PointGrey 社のサイト(<http://www.ptgrey.com/>)から、FlyCapture SDK をダウンロードします。(ユーザ登録が必要です。)

②

③ パッケージをインストールします。

```
$ sudo apt-get install libglademm-2.4-1c2a libgtkglextmm-x11-1.2-dev libserial-dev
```

④ ダウンロードしたアーカイブを展開します。

```
$ tar xvfz flycapture2-2.6.3.4-amd64-pkg.tgz
```

⑤ インストーラを起動します。

```
$ cd flycapture2-2.6.3.4-amd64/
```

```
$ sudo sh install_flycapture.sh
```

⑥ 以下が表示されるのでキーボードで「y」と入力します。

This is a script to assist with installation of the FlyCapture2 SDK.

Would you like to continue and install all the FlyCapture2 SDK packages?

(y/n)\$ y

以下が表示されるのでキーボードで「y」と入力します。

...

Preparing to unpack updatorgui-2.6.3.4_amd64.deb ...

Unpacking updatorgui (2.6.3.4) ...

updatorgui (2.6.3.4) を設定しています ...

Processing triggers for man-db (2.6.7.1-1ubuntu1) ...

Would you like to add a udev entry to allow access to IEEE-1394 and USB hardware?

If this is not ran then your cameras may be only accessible by running flycap as sudo.

(y/n)\$ y

Autoware

以下の手順で Autoware を入手し、ビルドおよびインストールを行います。

github から最新を入手する場合

```
$ git clone https://github.com/CPFL/Autoware.git  
$ cd Autoware/ros/src  
$ catkin_init_workspace  
$ cd ../  
$ ./catkin_make_release  
$ source devel/setup.bash
```

アーカイブを使用する場合

```
$ wget http://www.pdsl.jp/app/download/1039444574/Autoware-beta.zip  
$ unzip Autoware-beta.zip  
$ cd Autoware-beta/ros/src  
$ catkin_init_workspace  
$ cd ../  
$ ./catkin_make_release  
$ source devel/setup.bash
```

AutowareRider

AutowareRider は、ROS PC で動作する Autoware をタブレット端末から操作するための、Knight Rider に似た UI を持った、Android アプリケーションです。

以下の URL から APK ファイルを入手し、インストールを行います。

本体

- ✧ AutowareRider.apk
<https://github.com/CPFL/Autoware/blob/master/ui/tablet/AutowareRider/AutowareRider.apk>

経路データ生成アプリケーション

- ✧ AutowareRoute.apk
<https://github.com/CPFL/Autoware/blob/master/ui/tablet/AutowareRoute/AutowareRoute.apk>

CAN データ収集アプリケーション

- ✧ CanDataSender.apk
<https://github.com/CPFL/Autoware/blob/master/vehicle/general/android/CanDataSender/bin/CanDataSender.apk>
- ✧ CanGather.apk
<https://github.com/CPFL/Autoware/blob/master/vehicle/general/android/CanGather/apk/CanGather.apk>
- ✧ CarLink_CAN-BT_LS.apk
https://github.com/CPFL/Autoware/blob/master/vehicle/general/android/CarLink/apk/CarLink_CAN-BT_LS.apk
- ✧ CarLink_CANusbAccessory_LS.apk
https://github.com/CPFL/Autoware/blob/master/vehicle/general/android/CarLink/apk/CarLink_CANusbAccessory_LS.apk

CanGather は APK ファイル以外に、設定ファイルを用意する必要があります。

詳細は、以下の URL を参考にしてください。

<https://github.com/CPFL/Autoware/tree/master/vehicle/general/android#cangather-%E3%81%AE%E5%A0%B4%E5%90%88>

canlib

kvaser のサイト(<http://www.kvaser.com/downloads>) の "Kvaser LINUX Driver and SDK" よりソースコード linuxcan.tar.gz を入手し、以下の手順でインストールを行います。

```
$ tar xzf linuxcan.tar.gz  
$ cd linuxcan  
$ make  
$ sudo make install
```

SSH の公開鍵の作成

pos_db は、SSHを介してデータベースにアクセスします。その際、パスフレーズなしの SSH 鍵を使用します。

そのため、pos_db を使用する場合は、データベースサーバ用の SSH 鍵を以下の手順で作成し、SSH 公開鍵をデータベースサーバに登録する必要があります。

① SSH 鍵の作成方法

以下のコマンドを実行して鍵を作成します。

```
$ ssh-keygen -t rsa
```

その際、パスフレーズは空にして作成します。

(文字列を入力せずに Enter キーを押下する)

DSA を使用する場合は -t dsa と指定します。

② SSH 公開鍵をデータベースサーバに登録する

作成した SSH 公開鍵を以下のコマンドでサーバーにコピーします。

```
$ ssh-copy-id -i ~/.ssh/id_rsa.pub posup@db3.ertl.jp
```

("posup" はユーザ名、"db3.ertl.jp" はデータベースサーバ名)

その際にパスワードを聞かれるので適宜入力してください。

チャプター 6

6. 用語

用語	解説
3次元地図	一般なカーナビゲーションなどで使われている2次元の地図とは違い、道路網辺に設置されている立体などを含めてさまざまな情報が取り込まれた地図。
Autoware	[Autoware]ROS上で動くオープンソースソフトウェア
AutowareRider	[Autoware]ROS PCで動作するAutowareを、タブレット端末から操作するためのAndroidアプリケーション。TVドラマ「ナイトライダー」に似たUIを持つ。
AutowareRoute	[Autoware]MapFan SDKで実装された、経路データ生成のためのAndroidアプリケーション。
CAN	<i>Controller Area Network</i> 相互接続された機器間のデータ転送に用いられる規格。ドイツのOSCH社が車載ネットワークとして提唱し、ISO11898およびISO11519として標準化された。車載ネットワークの標準となっている。自動車の専用用機器や工業機器で採用されている。
catkin	[ROS]独自のロードシステム
CUDA	<i>Compute Unified Device Architecture</i> NVIDIA社が提供する、GPUを使った汎用計算プラットフォームとプログラミングモデル。
DMI	<i>Distance Measuring Instrument</i> 走行距離計
DPM	<i>Deformable Part Models</i> 物体検出手法
FlyCapture SDK	PointGrey社のカメラを制御するためのSDK。
FOT	<i>Field Operational Tests</i> 実際の運転環境下で、交通環境・ドライバ操作・車両動作の3者を長時間複数の車両運転者に跨って観測することで、運転を支える技術や知識の有用性を統計的に検証すること。新しい自動車開発方針法論の全般や環境など社会と深くかかわる自動車高高度化に要である。

用語	解説
GNSS	<i>Global Navigation Satellite System</i> 衛星測位システム。
IMU	<i>Inertial Measurement Unit</i> 慣性計測装置。角速度加速度を測る装置。
KF	<i>Kalman Filter</i> ルンゲルタ 誤差のある離散な観測情報と、刻々と時間変化する量から目標の位置を推定する手法
LIDAR	<i>Laser Imaging Detection and Ranging</i> /レーザスキャナ/レーザレーダ レーザーをパルス状に発光し、その反射に対する散乱光を測定、遠距離 00m 程度 ある物体への距離を計したり、その物体の立形ジング(形状)や特性を測定することができる機器。
Message	[ROS]ノード同士が信する際のデータ構造。
NDT	<i>Normal Distributions Transform</i> 位置推定手法
Node	[ROS]単一の能を提供するプロセス
Oculas	
Odometer 輪の	車輪回角と回転角速度を算て位置を推定する手法。
OpenCV	<i>Open source Computer Vision library</i> コンピュータビジョンを扱うための画像処理ライブラリ
Point Cloud	3次元空間の点の集合(点群)データ。直交座標(x, y, z)で表現される。物体の立体的な形状をこの点群を用て現できる。
Qt	アプリケーション・ユーザ・インターフェースのフレームワーク。
Quick Start	[Autoware]Autowareを起動した時に最初に表示されるGUIの最初のタブ。
ROS	ロボットソフトウェア開発のためのソフトウェアフレームワーク。ハードウェア抽象化や低レベルデバイス制御、よく使われる機能の実装、プロセス間通信、パッケージ管理などの機能を提供する。
ROS PC	ROSとAutowareをインストールしたパソコン。
rosbag	[ROS]データロギングツール。拡張子は bag
rqt	[ROS]QtベースのGUIソフト開発ツール
Runtime Manager	[Autoware]Autowareのデベロッパー用インターフェース。Autoware起動時に表示される。
rviz	[ROS]データとソフトウェア状態の視覚化ツール。
SLAM	<i>Simultaneous Localization and Mapping</i> 自己位置と環境地図作成を同時に進行する
TF	[ROS]座標変換ライブラリ。

用語	解説
Topic	[ROS]メッセージを送受信する先。メッセージの送信を「Publish」、受信を「Subscribe」と呼ぶ。
way point	経路に定された1 m間隔の目標
キャリブレーション	カメラに投影された点と3次元空間中の位置を合わせるため、カメラのパラメータを求める処理
高精度図	MMS (Mobile Mapping System) を利用して得したデータを用いて生成した高精度な三次元デジタル地図データ。従来の2次元データによる線表現に対し、曲線を表現できる3次元データであるとともに精、5センチ以内という高い精度を持つ。
センサ・フュージョン	複数のセンサ情報を組合せて、位置や姿勢をより正確に算出するなど、高度認識機能を実現する手法
ベクタマップ	ベクターで道路の地形情報を表現したG I S (Global Information System) データ。道路の地形情報はジオメトリで扱われ、このジオメトリのベクターデータで構成されている。
メッセージパッシング	1つ以上の受信者に対して送信者がデータを配信するプロセス間通信の方式

7. 関連文書

Autoware

<http://www.pdsl.jp/fot/autoware/>

AutowareRider

- ✧ 本体

AutowareRider.apk

<https://github.com/CPFL/Autoware/blob/master/ui/tablet/AutowareRider/AutowareRider.apk>

- ✧ 経路データ生成アプリケーション

AutowareRoute.apk

<https://github.com/CPFL/Autoware/blob/master/ui/tablet/AutowareRoute/AutowareRoute.apk>

- ✧ CAN データ収集アプリケーション

CanDataSender.apk

<https://github.com/CPFL/Autoware/blob/master/vehicle/general/android/CanDataSender/bin/CanDataSender.apk>

CanGather.apk

<https://github.com/CPFL/Autoware/blob/master/vehicle/general/android/CanGather/apk/CanGather.apk>

CarLink_CAN-BT_LS.apk

https://github.com/CPFL/Autoware/blob/master/vehicle/general/android/CarLink/apk/CarLink_CAN-BT_LS.apk

CarLink_CANusbAccessory_LS.apk

https://github.com/CPFL/Autoware/blob/master/vehicle/general/android/CarLink/apk/CarLink_CANusbAccessory_LS.apk

CUDA

http://www.nvidia.com/object/cuda_home_new.html

<http://www.nvidia.co.jp/object/cuda-jp.html>

<http://docs.nvidia.com/cuda/cuda-getting-started-guide-for-linux/>

<http://developer.nvidia.com/cuda-downloads>

FlyCapture SDK

<http://www.ptgrey.com/flycapture-sdk>

OpenCV

<http://opencv.org/>

<http://opencv.jp/>

<http://sourceforge.net/projects/opencvlibrary/>

Qt

<http://www.qt.io/>

<http://qt-users.jp/>

ROS

<http://www.ros.org/>

Ubuntu Japanese Team

<https://www.ubuntulinux.jp/>

Ubuntu

<http://www.ubuntu.com/>

Velodyne ドライバ

<https://github.com/ros-drivers/velodyne>

デモ関連

- ✧ デモ用の launch ファイルを生成するスクリプト

http://db3.ertl.jp/autoware/sample_data/my_launch.sh

- ✧ デモで使うデータ(守山地区の地図・キャリブレーション・経路)

http://db3.ertl.jp/autoware/sample_data/sample_moriyama_data.tar.gz

- ✧ ROSBAG データ

http://db3.ertl.jp/autoware/sample_data/sample_moriyama_150324.tar.gz

注) この ROSBAG データには画像情報が含まれていないため、物体検出(Detection)はできません